КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій
(факультет)

управління проєктами
(кафедра)

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО АТЕСТАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

на тему: «Управління проєктом розробки 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах»

АРТЕМЕНКО АРТЕМ ОЛЕГОВИЧ
(прізвище, ім’я та по батькові студента повністю)

Київ 2024 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій
(факультет)
управління проєктами
(кафедра)

ЗАТВЕРДЖУЮ
Завідувач кафедри УП
Бушуєв С.Д.

_____

„___” __ _____ ___2024 року

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО АТЕСТАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

на тему: «Управління проєктом розробки 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах»

АРТЕМЕНКО АРТЕМ ОЛЕГОВИЧ
(прізвище, ім’я та по батькові студента повністю)

Виконав: студент 4-го курсу, групи УП-41 .

Спеціальності: 126 «Інформаційні системи та технології» .
(шифр і назва напряму підготовки, спеціальності)
Спеціалізація: «Управління проєктами»

Артеменко А.О.
(прізвище та ініціали)

Керівник к.т.н, доцент Оберемок І.І.
(прізвище та ініціали)

Рецензент _____.
(прізвище та ініціали)

Київ 2024 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет: автоматизації і інформаційних технологій .
Кафедра: управління проєктами .
Освітній рівень: «бакалавр за ОП» .
Спеціальність: 126 «Інформаційні системи та технології» .
Спеціалізація: Управління проєктами .

ЗАТВЕРДЖУЮ
Завідувач кафедри УП
Бушуєв С.Д.
_____

„___” __ _____2024 року

З А В Д А Н Н Я
ДО ВИКОНАННЯ АТЕСТАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

_____ Артеменко Артем Олегович _____

1. Тема роботи: Управління проєктом розробки 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах.
затверджена наказом ректора КНУБА № 399/2 від «21» 02 2023 р.
2. Керівник роботи: Оберемок Іван Іванович, кандидат технічних наук, доцент кафедри управління проєктами.
3. Строк подання студентом роботи до захисту: червень 2024 року .
4. Зміст пояснювальної записки за розділами:
Р.1. Характеристика і аналіз предметної області та постановка задачі.
Р.2. Інформаційне та математичне забезпечення.
Р.3. проєктні рішення та розробка програмного продукту.
Р.4. Управління проєктом.
5. Інформаційні слайди:
С.1.

С.2.
С.3.
С.4.
С.5.
С.6.
С.7.
С.8.

6. Календарний план виконання АВР

Види робіт та їх зміст	Дата виконання
Р.1. Характеристика і аналіз предметної області та постановка задачі.	Травень 2024 р.
Р.2. Інформаційне та математичне забезпечення.	Травень 2024 р.
Р.3. Проєктні рішення та розробка програмного продукту.	Червень 2024 р.
Р.4. Управління проєктом.	Травень 2024 р.
Остаточне оформлення роботи	Червень 2024 р.
Попередній захист роботи на кафедрі	Червень 2024 р.

7. Консультанти розділів АВР

Розділ	Прізвище, ініціали та посада консультанта, представника комісії	дата	підпис
Ергономіка інформаційних технологій			
Прийом програмного продукту			

8. Дата видачі завдання: 2 лютого 2024 року

Керівник _____ Іван ОБЕРЕМОК
(підпис) (ім’я та прізвище)

Бакалавр _____ Артем АРТЕМЕНКО
(підпис) (ім’я та прізвище)

АНОТАЦІЯ

Артеменко А.О. Управління проєктом розробки 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах.

Атестаційна випускна робота бакалавра за спеціальністю: 126 «Інформаційні системи і технології», освітня програма «Управління проєктами». – Київський національний університет будівництва та архітектури. – Київ, 2024.

Атестаційна робота присвячена розробці 3D графіки для гри з тематикою виживання в скрутних умовах та створенню програмного продукту для інтеграції в програмне забезпечення Autodesk Maya. Розроблений програмний продукт дозволяє автоматизувати та забезпечити оптимізований процес розробки 3D графіки. В підсумку це забезпечить зручність для розробників.

Результатом виконання атестаційної роботи, були проаналізовані методи управління проєктом, створення трьовимірної графіки, розроблена інформаційна модель системи, UML-діаграма класів, розроблений плагін Autodesk Maya. Для управління проєктом був обраний метод Waterfall, ця методика дозволила виконувати усі поставлені задачі поетапно.

SUMMARY

Project management of 3D graphics development for a computer game with the theme of survival in difficult conditions.

Bachelor's thesis for a bachelor's degree in specialty: 126 “Information Systems and Technologies”, educational program “Project Management.” - Kyiv National University of Construction and Architecture - Kyiv, 2024.

The certification work is devoted to the development of 3D graphics for a game with the theme of survival in difficult conditions and the creation of a software product for integration into Autodesk Maya software. The developed software product allows to automate and provide an optimized process of 3D graphics development. As a result, it will provide convenience for developers.

As a result of the certification work, we analyzed the methods of project management, creation of three-dimensional graphics, developed an information model of the system, a UML class diagram, and developed an Autodesk Maya plugin. The Waterfall method was chosen to manage the project, as it allowed us to complete all the tasks in stages.

ЗМІСТ

ВСТУП

Управління проєктом розробки 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах є є актуальною темою. В сьогоднішніх реаліях, де ріст розвитку технологій набирає досить швидкі темпи, ігрова індустрія стає потужним засобом не лише для розваги, але й освіти і культурного впливу. Розробка якісних комп’ютерних ААА-проєктів з використанням новітніх методів створення 3D графіки сприяє підвищенню рівню технологічної грамотності, розвивається креативне мислення та залучує все більш людей до ІТ-сфери.

Для України ця тема має особливе значення. В умовах сучасних викликів, українськи розробники можуть використовувати свої ігрові проєкти для привернення уваги до важливих соціальних та екологічних проблем. Ігри з тематикою виживання в скрутних умовах можуть не тільки мати розважальний характер, але й навчати користувачів навичкам подолання різних труднощів, адаптації та виживання в складних умовах. Крім того, розвиток цього напрямку сприяє створенню нових робочих місць, залученню інвестицій та популяризації українського продукту на міжнародному рівні.

Цей дипломний проєкт присвячений розробці ефективної стратегії управління проєктом, яка включає всі етапи створення 3D графіки для гри, починаючи з концептуального дизайну до фінальних кроків створення. Особлива увага приділяється розробці програмного продукту для програмного забезпечення Autodesk Maya, що дозволить оптимізувати та скоротити терміни розробки кінцевого продукту.

РОЗДІЛ 1 Характеристика і аналіз предметної області та постановка задачі.

1.1 Характеристика діючої системи управління та проєктування

Аналіз сучасного стану управління та проєктування.

На сучасному етапі розвитку ігрової індустрії, управління та проєктування вимагає вдосконалення з урахуванням зростаючих технічних можливостей та ринкових вимог. Системи управління проєктами, такі як:

1.Agile - це сукупність підходів і моделей поведінки, орієнтованих на використання ітеративної розробки, time boxes (часових рамок), динамічне формулювання вимог і забезпечення реалізації ПЗ в результаті взаємодії всередині високо самоорганізованої робочої групи із фахівців різних профілів.

2.Scrum - це фреймворк управління, згідно з яким одна чи декілька кросфункціональних команд створюють продукт інкрементами, тобто, поетапно. В команді може бути близько семи людей».

3.Waterfall - традиційна, найпоширеніша та логічна методологія управління проєктами. Водоспадна модель передбачає послідовне проходження процесу, розбитого на стадії або етапи. Зазвичай її застосовують до проєктів, які можуть бути поділені на послідовні логічні частини. До того ж жоден етап не може бути виконаний раніше за попередній. Логіку такого проєкту легко зрозуміти та описати.

Ці системи широко використовуються для планування, виконання та контролю проєктів у галузі розробки відеоігор, тому проаналізувавши сучасні методи управління можна зробити вибір на методології управління ‘‘Waterfall’’, цей метод ідеально підходить для управління проєктом розробки 3D графіки для гри з тематикою виживання в скрутних умовах та являється актуальним на сьогоднішній час.

Ось декілька основних причин чому саме обраний метод підходить для проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах:

1.Наявність чітко визначених вимог та завдань - в рамках нашого проєкту присутні чіткі та визначені етапи та рішення щодо поліпшення продуктивності

процесу розробки. Детально описані завдання будуть показані у четвертому розділі дипломного проєктування.

2. Зрозумілі вимоги та очікування – в даному випадку специфікації створення 3D моделей визначені на початковому етапі.

3. Легка відстежуванність прогресу – кожен етап створення продукції, а саме 3D моделей має конкретну ціль та результат.

4.Зменшення змін у процесі – одною з основних особливостей даного проєкту є те, що важливі рішення будуть прийняті на самому початку, тому це мінімізує ризики значних змін та затримок під час розробки.

5.**Фіксований бюджет та вимоги** – проєкт створення 3D графіки для гри з тематикою виживання в скрутних умовах має на початку фіксованих бюджет та вимоги, більш детально про ці аспекти буде описано у четвертому розділі.

Особливості управління проєктом розробки 3D графіки

Управління проєктом розробки 3D графіки має свої особливості, зумовлені високим ступенем технічної складності та творчого характеру завдань.Основні складові системи управління проєктом включають планування, виконання, контроль та оцінку результатів.

Процеси проєктування 3D графіки

Процеси проєктування 3D графіки включають створення концепційних зображень, моделювання об'єктів, текстурування та розробки індивідуальних рішень. Кожен з цих етапів вимагає спеціалізованих інструментів та кваліфікованих фахівців. Застосування інтегрованих середовищ розробки (IDE) та графічних редакторів, таких як Autodesk Maya, Blender, Adobe Photoshop тощо, сприяє ефективному виконанню завдань та спільній роботі команди.

Виклики управління та проєктування

Одним із головних викликів управління проєктом розробки 3D графіки є забезпечення відповідності графічного контенту вимогам проєкту, з урахуванням обмежень термінів та бюджету. Також важливо забезпечити зручність спілкування та

співпраці між членами команди, щоб уникнути непорозумінь та забезпечити однорідність розуміння цілей та завдань проєкту.

Інструменти управління

Управління проєктом розробки 3D графіки може бути полегшено за допомогою використання спеціалізованих програмних засобів, таких як системи контролю версій. Наприклад:

1.Git – це розподілена система контролю версій, яка дозволяє відстежувати історію розробки ПЗ і спільно працювати над складними проєктами з будь-якої точки світу.

Також досить важливими ПЗ для спільного контролю ресурсів є:

1.Dropbox - це місце, де зібрано весь вміст команди розробників, де є можливість користуватись широким спектром інструментів, відкидувати все зайве, щоб зосередитися лише на найважливіших справах.

2.Google Drive - платформа для обміну файлами й особисте хмарне сховище – Google. Інтегрований набір безпечних хмарних додатків на основі технологій Google AI для ефективної роботи й співпраці.

Спеціалізовані платформи для управління проєктами, наприклад:

1.Jira — це система управління проєктами, яка дозволяє закривати майже всі завдання PM-а в рамках одного інструмента: від планування до контролю процесів та результатів. Є комплексом з IT-рішень від компанії Atlassian.

2. **Trello** - це візуальний інструмент, що дає змогу команді керувати різноманітними проєктами й робочими процесами та відстежувати виконання завдань.

3. Microsoft Project - система управління проєктами, розроблена корпорацією Microsoft. Microsoft Project створений, щоб допомогти менеджерові проєкту в розробці планів, розподілі ресурсів за завданнями, відстежуванні прогресу і аналізі обсягів робіт.

Використання цих інструментів дозволяє забезпечити ефективне планування, співпрацю та контроль над процесами розробки.

1.2Аналіз області застосування об'єкта проєктування

Проєкт розробки 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах відноситься до ігрової індустрії, що є однією з найбільш динамічних та швидкозмінних галузей розвитку програмного забезпечення. Він знаходить широке застосування у відомому жанрі ігор про виживання, де гравцям доводиться протистояти різноманітним небезпекам та використовувати обмежені ресурси для того, щоб вижити в скрутних умовах.

1.2.1 Досягнуті можливості на даний момент часу

Ігрова індустрія є суцільним полем новаторства та технологічного прогресу. Швидкий темп розвитку апаратного забезпечення та програмного забезпечення надає безмежні можливості для реалізації новаторських ідей у галузі геймдеву. Графіка у відеоіграх стає все більш фотореалістичною, динамічною та іммерсивною завдяки новітнім технологіям обробки графіки, таким як фізично засноване освітлення, техніки реалістичної анімації та високоякісні текстури.

Однією з основних сфер технологічного прогресу в ігровій індустрії є розвиток графічних технологій. Розробники активно використовують новітні технології, такі як рейтрейсінг, шейдери, HDR, фізично засноване рендеринг тощо, для досягнення максимальної реалістичності та деталізації візуального вигляду ігор. Реалістична 3D графіка відіграє ключову роль у створенні іммерсивного досвіду для гравців. За допомогою передових графічних движків:

1.Unreal Engine - інструмент для створення ігор і сцен із використанням 3D-моделей, розроблений компанією Epic Games. Ця програма призначена для створення ігор і симуляцій. Unreal Engine надає засоби для створення дивовижних графічних ефектів: високоякісне освітлення, рендеринг і підтримка віртуальної реальності.

2.Unity - це ігровий рушій, який використовується для створення 2D і 3D ігор для різних платформ, включно з персональними комп'ютерами, мобільними пристроями, консолями і навіть віртуальною реальністю.

3.Cryengine - це ігровий рушій розроблений німецькою компанією Crytek, який вперше був використаний у відеогрі Far Cry. Він був створений як технологічна демонстрація GeForce 3 від Nvidia. Ubisoft же має модифіковану версію CryEngine у грі Far Cry, який називається «Dunia Engine».

Також велику роль відіграють сучасні редактори тривимірної графіки, які володіють широкою функціональністю 3D-анімації, моделювання та візуалізації:

1.Autodesk Maya - застосунок, графічний редактор, для моделювання тривимірних об'єктів, анімації, композитингу та візуалізації (за допомогою підімкнутих систем рендерингу). В даний час є стандартом для розробки 3D графіки для кіно і телебачення.

2.Blender - це повністю інтегрований комплект створення 3D, який пропонує широкий діапазон основних засобів.

3. 3ds Max - тривимірний графічний редактор, повнофункціональний професійний застосунок, система для створення і редагування об'єктів та створення візуалізацій, розроблена компанією Autodesk. Містить найсучасніші засоби для архітекторів, дизайнерів, художників і фахівців в області мультимедіа.

4. Plasticity — це новий софт для 3D-моделювання на основі NURBS, яке розробляється спеціально для 3д-художників та дизайнерів.

5. ZBrush – програма для 3D-моделювання, створена компанією «Pixologic».Відмінною особливістю цього ПЗ є імітація процесу «ліплення» тривимірної скульптури, посиленого двигуном тривимірного рендерингу в реальному часі, що спрощує процедуру створення необхідного тривимірного об'єкта

6. Fusion 360 - це комплексний хмарний CAD/CAE/CAM інструмент для промислового дизайну та машинобудівного проєктування. Він поєднує в собі найкраще, що можна було взяти від Inventor, Alias, Simulation та інших програмних продуктів Autodesk, щоб створити унікальне середовище, яке з легкістю можна пристосувати під себе і яке дозволить спроєктувати практично все, що ви можете уявити.

7. Substance 3D Painter – це програмне забезпечення для 3D-малювання, яке дає користувачам змогу текстурувати та додавати матеріали безпосередньо до 3D-сіток у режимі реального часу.

Використовуючи наведене вище ПЗ розробники можуть створювати візуальні ефекти, деталізовані світи та живі персонажі та впроваджувати індивідуальні рішення.

Такі технологічні досягнення роблять ігри більш привабливими для гравців, але водночас ставлять перед розробниками нові виклики, пов'язані з оптимізацією продуктивності та забезпеченням сумісності з різними платформами.

Результати застосування досягнутих можливостей в іграх про виживання

Ігри з тематикою виживання в скрутних умовах відображають реалістичні сценарії, де гравцям доводиться боротися за виживання у ворожому середовищі. Це може бути постапокаліптичний світ, дикий природний ландшафт або інші несприятливі умови. Гравець стикається з різноманітними викликами, такими як пошук їжі, будівництво притулку, захист від ворогів та природних катастроф.

Одним з результатів застосування досягнутих можливостей використання новітніх технологій призвела до зростання уваги до атмосфери та амбієнту, це дозволило розробникам створювати набагато реалістичніші ігрові світи.

Також важливо відмітити, перехід до віртуальної та доповненої реальності. Розвиток AR та VR технологій за допомогою сучасного програмного забезпечення призвело до зростання попиту на тематику ігор виживання.

Одні з найуспішніших проєктів створення 3D графіки в ігровій індустрії :

1.Electronic Arts (EA): Один з найбільших видавців комп'ютерних ігор у світі.Випускає широкий спектр ігор різних жанрів, включаючи ігри з тематикою виживання, такі як "The Sims", "Dead Space" та "Plants vs. Zombies".

2.Ubisoft: Французька компанія, що спеціалізується на розробці та виданні відеоігор.Випускає серії ігор з тематикою виживання, такі як "Far Cry" та "Tom Clancy's The Division".

3.Bohemia Interactive: Чеська студія, розробник серії ігор "Arma" та "DayZ", які відомі своєю реалістичністю та виживальним геймплеєм.

1.2.2 Ідентифікація проблемних моментів

Розробка реалістичної 3D графіки для ігор про виживання також стикається з технологічними викликами. Високі вимоги до продуктивності та ефективного використання ресурсів обмежують можливості розробників. Оптимізація графіки для різних платформ, від ПК до мобільних пристроїв, вимагає розробки складних алгоритмів та стратегій управління ресурсами. Тому буде взято до уваги деякі проблемні моменти:

1.Складність реалізації високоякісної графіки:

Однією з основних проблем у розробці 3D графіки для ігор є складність досягнення високоякісної графіки. Вимагається набагато більше часу та зусиль для створення реалістичних текстур, моделей та ефектів, що може сповільнювати процес розробки та збільшувати витрати на продукцію.

2.Оптимізація продуктивності для різних платформ:

Ще однією важливою проблемою є оптимізація продуктивності графічного контенту для різних платформ. Оскільки ігри часто випускаються на різні пристрої, такі як ПК, консолі та мобільні пристрої, необхідно забезпечити оптимальну продуктивність для кожної з них. Це може бути особливо складним завданням при розробці для мобільних пристроїв, де обмежені ресурси можуть стиснути можливості відтворення високоякісної графіки.

3. Ризик недоліків у продукції:

Існує ризик недоліків у продукції, що може виникнути при розробці складної 3D графіки. Технічні проблеми, помилки у коді, артефакти графічного движку та інші непередбачені фактори можуть призвести до появи багів та вплинути на якість гри. Вирішення цих проблем може вимагати додаткового часу та ресурсів, що може затримати випуск гри та підвищити її витрати.

Отже, область застосування проєкту розробки 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах представляє собою високотехнологічне поле інновацій та творчості, де кожна деталь графіки має велике значення для створення захоплюючого іммерсивного досвіду для гравців.

1.2.3 Визначення мети дипломного проєктування

Проаналізувавши проблемні моменти були визначені основні недоліки у розробці продукту, тому в межах дипломного проєктування пропонується наступна мета автоматизованого проєктування:

Основна мета автоматизованого проєктування: Розробка якісного продукту, а саме 3D графіки шляхом впровадження власних розробок та інтеграцій в актуальне програмне забезпечення.

Опираючись на мету очікується одержати в ході дипломного проєктування:

- **Рішення щодо скорочення строків розробки та вирішення поставлених задач.**
- **Підвищення якості та технічного рівня результатів розробки.**
- **Якісний план та контроль проєкту.**

Головні завдання автоматизованого проєктування та проєкту:

1.Впровадження власних рішень та розробок: Проведення поетапного аналізу та впровадження індивідуальних рішень для підвищення ефективності створення продукту.

2.Розробка інтеграцій в програмне забезпечення: Початок етапу розробки автоматизованого функціоналу який надасть широкий спектр інструментарію і в результаті забезпечить швидкий процес розробки та скоротить терміни вирішення задач для створення 3D графіки для гри з елементами виживання в скрутних умовах.

2.Створення реалістичного віртуального світу: Розробка деталізованих і іммерсивних віртуальних локацій, що відображають скрутні умови виживання, з урахуванням фізичних законів та особливостей оточуючого середовища.

3.Моделювання персонажів та ворожих об'єктів: Створення реалістичних 3D моделей головного героя, ворогів та інших об'єктів з якими персонаж буде мати взаємодію, що допоможе підвищити іммерсивність та емоційну залученість гравців.

4.Текстурування та анімація: Розробка реалістичних текстур для об'єктів та персонажів, а також створення плавних та реалістичних анімацій.

5.Оптимізація продуктивності та сумісності: Забезпечення оптимальної продуктивності гри на різних платформах (ПК, консолі, мобільні пристрої), а також забезпечення сумісності з різними конфігураціями обладнання.

6.Тестування та виправлення помилок: Проведення комплексного тестування гри для виявлення та виправлення помилок, а також забезпечення стабільної та безперебійної роботи.

Отже, дипломне проєктування спрямоване на створення високоякісної 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах, а також розробки власних рішень для забезпечення продуктивності.

1.3 Основні цілі автоматизованого проєктування

У контексті розробки проєкту зі створення комп'ютерної гри з тематикою виживання в скрутних умовах, автоматизоване проєктування виступає як ключовий елемент для досягнення успіху проєкту. Застосування автоматизованих інструментів та методів дозволяє оптимізувати процеси розробки та забезпечити ефективне вирішення завдань. У цьому відомстві розглянемо основні цілі автоматизованого проєктування та його вплив на розробку комп'ютерної гри.

Отже, деякі з основних цілей автоматизованого проєктування:

1.Зменшення трудомісткості проєктних рішень та планування: Основною метою автоматизованого проєктування є зменшення зусиль, які необхідні для розробки стратегічних та тактичних рішень у процесі розробки гри. Шляхом використання відповідних інструментів можна автоматизувати процеси планування та вирішення ключових завдань проєкту.

2.Скорочення строків вирішення задач: Автоматизовані методи дозволять ефективно виявляти та вирішувати проблеми, що виникають під час розробки гри, що призводить до скорочення часу, потрібного для реалізації проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах.

3.Зменшення собівартості проєктування та виготовлення: Використання автоматизованих інструментів допомагає знизити витрати на розробку гри з тематикою виживання в скрутних умовах, шляхом оптимізації використання ресурсів та прискорення процесів реалізації.

4.Підвищення якості та техніко-економічного рівня результатів проєктування: Автоматизація проєктування дозволяє покращити якість розробки гри шляхом зменшення помилок та підвищення точності прийняття рішень.

5.Зменшення витрат на натурне моделювання та тести: Використання автоматизованих методів дозволяє ефективно використовувати віртуальні та симуляційні середовища для тестування та перевірки рішень, що допомагає зменшити витрати на фізичне моделювання та іспити.

1.4 Дерево цілей проєктування

Дерево цілей для проєктування комп'ютерної гри з тематикою виживання в скрутних умовах виглядає наступним чином:

1.Розробка інтеграції в програмне забезпечення.
1.1. Аналіз наявних проблем у процесі розробки та підготовка до розробки.
1.1.2. Розробка програмного коду.
Розробка програмного коду
Розробка інтерфейсу
1.1.3. Тестування програми
Інтеграція в програмне забезпечення
Тестування

1.2. Початок створення персонажів та об'єктів з використанням власного інтегрованого програмного плагіну.
1.2.1. Створення моделі головного персонажу
проєктування реалістичних характеристик та анімацій
Розробка зовнішнього вигляду та екіпірування
Текстурування та деталізація
1.2.2. Розробка моделей ворожих об'єктів
проєктування різноманітності ворожих персонажів та істот
Створення анімацій та реакцій на взаємодію з головним героєм
Визначення інтелектуальних та фізичних властивостей ворогів
1.2.3. Анімація персонажів та об'єктів
Створення реалістичних та плавних рухів персонажів
Реалізація анімацій для взаємодії персонажів з оточуючим середовищем
Відтворення різноманітних ефектів та анімаційних сцен

1.3. Текстурування та освітлення
1.3.1.Створення текстур для об'єктів та персонажів
Підготовка текстур високої роздільної здатності
Адаптація текстур до різних локацій та умов освітлення
1.3.2. Оптимізація освітлення віртуального світу
Реалізація різних джерел світла
Створення тіней та об'ємного освітлення

Оптимізація освітлення для досягнення плавного та реалістичного відображення

1.4. Оптимізація продуктивності та сумісності

1.4.1. Підтримка різних платформ (ПК, консолі, мобільні пристрої)

Адаптація графічного контенту до можливостей різних платформ

Виправлення помилок та оптимізація продуктивності для кожної платформи

Тестування гри на різних пристроях для виявлення сумісності

1.4.2. Оптимізація продуктивності гри на різних пристроях

Використання різних рівнів деталізації для підтримки різних пристроїв

Оптимізація алгоритмів рендерингу для забезпечення плавної гри на слабших пристроях

Підтримка різних роздільних здатностей та екранних форматів для різних пристроїв

1.5 Аналіз поставленого завдання

1.Опис завдання:Поставлене завдання полягає у розробці 3D графіки для гри з тематикою виживання в екстремальних умовах з використанням програмних рішень.

2.Аналіз цільової аудиторії ринку:Гравці, які цікавляться іграми з тематикою виживання, шукають захоплюючий та реалістичний ігровий досвід. Вони очікують деталізованого світу з відтворенням різних атмосфер та умов виживання.

3.Технічна оцінка проєкту:Розробка 3D графіки та програмних рішень вимагає великих обчислювальних ресурсів та спеціалізованого програмного забезпечення для моделювання.

4. Ідентифікація труднощів та ризиків:Створення деталізованих локацій та персонажів може зайняти значний час та вимагати великої кількості ресурсів. Оптимізація гри для різних платформ може стикатися з труднощами у забезпеченні стабільності та продуктивності на різних пристроях.

Загальний аналіз показує, що успішна реалізація цього завдання вимагає великих зусиль, спеціалізованих знань та досвіду в різних галузях розробки комп'ютерних ігор.

1.5.1 Огляд відомих досліджень з теми дипломної роботи

В реалізації будь-якого проєкту ключовою складовою є управління процесами розробки та координація робіт. Особливо це стосується розробки великих проєктів, таких як комп'ютерні ігри з використанням 3D графіки, особливо з тематикою виживання в скрутних умовах. В цьому підрозділі ми розглянемо огляд відомих досліджень з цієї теми, звертаючи особливу увагу на публікації в наукових журналах та періодичних виданнях.

Ретельний аналіз цих досліджень дозволить зрозуміти сучасний стан справ у галузі управління проєктом розробки відеоігор і з'ясувати можливості використання відомих рішень або необхідність проєктування оригінальних підходів.

1.Публікація: "Effective Project Management Strategies for Developing 3D Graphics in Survival-themed Video Games"

Видання: Journal of Game Development

Ця стаття пропонує стратегії ефективного управління проєктом розробки 3D графіки в виживальних відеоіграх. Автори досліджують ключові етапи проєктування та визначають оптимальні підходи до вирішення складнощів у розробці.

2. Публікація: "Innovative Techniques for Integrating 3D Graphics into Survival-themed Video Games"

Видання: ACM Transactions on Graphics

У цій статті описані новаторські техніки інтеграції 3D графіки в виживальні відеоігри. Автори вивчають технічні аспекти розробки та надають поради щодо оптимального використання графічних можливостей.

3.Публікація: "The Evolution of Video Game Graphics: From Pixels to Realism"

Сайт:Medium.com

Автор: Atticus Madison

Стаття "The Evolution of Video Game Graphics: From Pixels to Realism" ретельно досліджує еволюцію графіки в комп'ютерних іграх з початкових піксельних зображень до вражаючого реалізму, який ми бачимо сьогодні. Вона розглядає кожен етап розвитку графіки, починаючи з простих 2D графічних ігор

і переходячи до потужних 3D середовищ та високоякісної графіки в епоху високої чіткості.

4.Публікація: Crafting Realities
The Past, Present, and Future of Open World Survival Craft Games

Сайт: Medium.com

Автор: JohnTheKraken

Стаття "Crafting Realities: The Past, Present, and Future of Open World Survival Craft Games" пропонує огляд жанру "Відкритий Світ Виживання та Ремесло" (OWSC), що описується як стиль гри, в якому гравці мають необмежену можливість дослідження, механіки ремесла та постійні загрози навколишнього середовища. Стаття розглядає етапи розвитку цього жанру, його визначальні особливості, ключові ігри та їх вплив на гравців і галузь в цілому.

Також розглядаються перспективи жанру у майбутньому, включаючи можливості використання технологій віртуальної та розширеної реальності, розвиток ігрових механік та обробки штучного інтелекту. В цілому, стаття відзначає важливість жанру та його потенціал для майбутнього розвитку.

Отже, знайдені дослідження та публікації у наукових журналах та періодичних виданнях свідчать про актуальність теми та наявність інтересу до управління проєктами в розробці сурвайвл ігор з використанням 3D графіки. Можливість використання відомих рішень та практик зазначених у дослідженнях буде корисною для успішної реалізації проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах.

Однак, також важливо розглянути можливість проєктування оригінальних рішень, щоб створити унікальний геймплей та відповісти на унікальні вимоги проєкту.

1.5.2 Огляд схожих проєктів

Оскільки тема дипломної роботи стосується управління проєктом розробки 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах, давайте розглянемо кілька аналогічних проєктів:

1. Subnautica:
Тематика: Гравці розглядають підводний світ на планеті, де вони повинні виживати в різних умовах та боротися з небезпечними морськими істотами.

Графіка: Гра має вражаючу 3D графіку, що відтворює дивовижні морські пейзажі та істот.

Механіка гри: Гравці повинні виживати, збирати ресурси, розблоковувати нові технології та досліджувати підводний світ.

2. The Forest:
Тематика: Гравці виживають у лісі після авіакатастрофи, будують притулки, збирають ресурси та борються з місцевими істотами.

Графіка: Гра має реалістичну 3D графіку, яка допомагає створити напружену та жахливу атмосферу.

Механіка гри: Гравці повинні будувати укриття, вирощувати їжу, досліджувати навколишній ліс та змагатися з ворожими племенами.

3. DayZ:
Тематика: Гравці виживають у постапокаліптичному світі, де вони змушені шукати їжу, воду, одяг та зброю, уникати зомбі та інших гравців.

Графіка: Гра використовує вражаючу 3D графіку, яка передає атмосферу пустельного світу, де кожен куток приховує нову загрозу або можливість.

Механіка гри: Гравці повинні виживати в неблагополучних умовах, шукаючи ресурси, обороняючись від ворожих істот та взаємодіючи з іншими гравцями, щоб вижити в цьому безжальному світі.

1.5.3 Висновки щодо можливості використання відомих рішень або необхідності проєктування оригінальних рішень

1.Огляд відомих рішень:

Технологічні вдосконалення: Подібні проєкти, як " Subnautica", " The Forest", "DayZ", демонструють значний прогрес у галузі візуальних ефектів, геймплею та механіки виживання.

Розвиток спільноти: Ці ігри створили великі та активні спільноти гравців, що може бути корисним для обміну досвідом та ідеями під час розробки.

2.Можливість використання відомих рішень у проєкті управління створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Використання графіки та механіки гри: Відомі рішення нададуть важливі вказівки щодо того, як виглядає успішна гра у жанрі виживання.

Спільнота користувачів: Запозичення ідей з інших відомих проєктів може сприяти залученню уваги та підтримки спільноти гравців.

3. Необхідність проєктування оригінальних рішень у проєкті створення рення 3D графіки для гри з тематикою виживання в скрутних умовах:

Конкурентне середовище: Хоча відомі рішення можуть надати важливі вказівки, конкурентне середовище галузі вимагає оригінальних інновацій, щоб виділитися серед інших гравців на ринку.

Задоволення потреб цільової аудиторії: Розуміння унікальних потреб та бажань цільової аудиторії може вимагати створення унікальних рішень, що відповідають їхнім очікуванням та побажанням.

Отже, використання відомих рішень може служити важливим джерелом натхнення та вказівок, але для успішного конкурування на ринку необхідно також розробляти оригінальні інновації, що відповідають потребам цільової аудиторії та виокремлюють проєкт створення 3D графіки з тематикою виживання в скртуних умовах серед конкурентів.

1.6 Класифікація, основні характеристики об'єкта дослідження та формулювання постановки задачі

У цьому підрозділі проводиться класифікація та аналіз основних характеристик об'єкта дослідження, яким є проєкт управління розробкою 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах.

Глибокий розгляд основних рис гри дозволить краще зрозуміти її суть, механіку геймплею та особливості, що відрізняють її від інших представників жанру. Розглянуті аспекти визначають унікальність ігрового досвіду та відображають ключові елементи, які формують цей проєкт у цілісну та привабливу гру для гравців.

Жанр гри: Управління проєктом розробки 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах належить до категорії відкритого світу (open world) та виживання (survival).

Основні характеристики:

1.Відкритий світ: Гравці мають можливість досліджувати обширний інтерактивний світ без обмежень на лінійний прогрес.

2.Виживання: Гравці повинні боротися за виживання в неблагоприятних умовах, управляючи ресурсами, такими як їжа, вода, тепло, і зіткненнями з ворожими сила місцевого середовища.

3.Розвинена система крафту: Ігровий процес передбачає збір ресурсів та виготовлення з них інструментів, зброї, будівель та інших корисних предметів.

4.Динамічна атмосфера: Світ гри може змінюватися з часом, включаючи зміну погодних умов, денно-ночний цикл та інші фактори, що впливають на геймплей.

Ці характеристики визначають основні аспекти ігрового досвіду у проєкті, що досліджується, і визначають його унікальність серед інших ігор у жанрі.

1.7 Висновки

1. Проведений аналіз та характеристика діючої системи управління та проєктування.
2. Проведений аналіз застосування об'єкта проєктування.
3. Проведено дослідження досягнутих можоливостей.
4. Проведений аналіз результатів досягнутих можливостей.
5. Ідентифіковані проблемні моменти.
6. Визначена мета дипломного проєктування.
7. Визначені основні цілі дипломного проєктування.
8. Побудовано дерево цілей проєктування.
9. Проведений аналіз поставленого завдання.
10. Проведений детальний огляд досліджень з теми дипломної роботи .
11. Проведений аналіз схожих проєктів.
12. Зроблені висновки можливості використання відомих рішень та проєктування відомих рішень.
13. Проведений аналіз проєктування оригінальних рішень.
14. Класифіковані основні характеристики об'єкту дослідження.

РОЗДІЛ 2 Інформаційне та математичне забезпечення.

2.1 Вступ до другого розділу

У сучасному світі ігрова індустрія відіграє важливу роль як у розважальній, так і у виховній сферах суспільства. Швидкі та постійні зміни в цій галузі вимагають постійного вдосконалення та адаптації до сучасних технологій та вимог гравців. Однією з найбільш цікавих та популярних форм ігор є комп'ютерні ігри з тематикою виживання в скрутних умовах.

Одним з ключових аспектів, які визначають успіх таких ігор, є якісна та реалістична 3D графіка, яка відтворює оточення з вражаючою деталізацією та анімацією.

В рамках даного дипломного проєкту присвячено розробці 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах, де основною метою є управління процесом створення 3D оточення. Для досягнення успіху в цьому проєкті необхідно провести детальний аналіз предметної області, щоб зрозуміти основні вимоги до графічних об'єктів, які будуть створені, а також врахувати сучасні тенденції та стандарти розробки в ігровій індустрії.

У цьому розділі буде проведено аналіз актуального пайп лайну створення 3D оточення 3D графіки для гри з тематикою виживання. Описані аспекти будуть детально проаналізовані з метою визначення ключових вимог та варіантів їх реалізації та поліпшення у рамках даного проєкту.

Цей розділ буде важливим етапом у підготовці та плануванні процесу розробки трьохмірної графіки, оскільки він надасть базову інформацію та рекомендації для подальших етапів проєкту.

2.2 Мета дослідження предметної області

Метою дослідження предметної області в рамках даного дипломного проєктування, а саме створення 3D графіки для гри з тематикою виживання в скрутних умовах – аналіз актуальних методів створення продукту, детальний опис процесу розробки та виявлення її слабких сторін для надання майбутніх рішень щодо її поліпшення. Також буде проаналізовано та визначено саме ефективне програмне забезпечення.

Цей аналіз буде надавати чітке розуміння щодо планування майбутніх процесів розробки.

2.3 Виділення та аналіз об’єктів дослідження

В цьому пункті буде розглянуто та виділено основні об’єкти дослідження та наявні інструменти його створення та буде отримано базову конкретику для

подальшого детального аналізу та виявлення проблем в рамках дипломного проєктування.

Отже, одним з об’єктів дослідження нашої роботи – 3D модель, яка відповідає стандарту ігрової індустрії.

Game ready 3d model(Гейм реді 3D модель) – це центральний елемент у розробці ігор, яка впливає на візуальну привабливість та продуктивність гри. Простими словами 3D моделі в ігровій індустрії та в рамках теми дипломного проєктування можна класифікувати такими об’єктами:

1.Природні об'єкти:
Ландшафт:
Рельєф, різноманітність поверхні, природні об’єкти(трава, кущі, дерева, річки, озера, гори, фонтани, впадини)

2. Споруди та конструкції:
Будинки: Створення будинків різних типів та розмірів, включаючи котеджі, хижини та багатоквартирні будівлі.
Примітивні укриття: Розробка укриттів, таких як печери, намети та барлоги.
Вежі та бункери: Створення веж, які можуть служити для спостереження або оборони, а також бункерів для захисту від небезпек.

3. Персонажі та істоти:
Головний персонаж – герой яким керує гравець.
Ворожі істоти – істоти з якими буде стикатись головний персонаж під час проходження основного сюжету гри.
Тварини – різноманітні звичні тварини які зустрічаються у повсякденному житті людини.

4. Знаряддя та ресурси
Зброя та інструменти: Ножі, сокири, вогнепальна зброя, інструменти для виживання.
Елементи життєздатності головного персонажу: Їжа, вода, ліки, матеріали для виготовлення предметів.

5.Транспорт:
Транспортні засоби: автомобілі, мотоцикли, водні засоби транспорту і т.п.

В проєкті створення 3D графіки для гри на тематику виживання в скрутних умовах буде використовуватись полігональний метод моделювання об’єктів, тому потрібно розглянути та проаналізувати актуальний пайп-лайн створення

тривимірних об’єктів для виявлення слабких сторін методу задля майбутніх програмних рішень.

Отже, почнемо аналіз актуального методу створення об’єктів у сучасних ААА-проєктів:

1.Концептування – перший етап у створенні будь-якої тривимірної моделі незалежно від напрямку використання. Концептуальні зображення в кінцевому результаті відображає ідею та бачення фінального продукту.
В нашому проєкті переважна кількість концепт-артів будуть відноситись до post-apocalyptic style(постапокаліптичний стиль).

2.Розробка високо-полігональної(High Poly Modeling) моделі – після отримання концепт-арту починається розробка самої ідеї шляхом полігонального моделювання. Високополігональна модель включає у себе розробку форми та деталей об’єкту, тобто створення базової деталізованої геометрії.

3.Розробка низько-полігональної моделі(Low Poly Modeling) – низько-полігональна модель будується на основі високо-полігональної. Розробка низько-полігональної моделі зумовлена тим, що ця модель в майбутньому буде використовуватись у реальному часі, тому її потрібно оптимізувати, тобто зменшити кількість полігонів, що призведе до отримання оптимізованого базового продукту.

4.Розгортка UV (UV Unwrapping) – створює двовимірне зображення карти, яка відображає тривимірну модель. Цей процес дуже важливий та часовитратний, саме цей етап створення стане причиною впровадження індивідуальної розробки програмної інтеграції для дипломного проєктування.

5.Запікання карт(Baking Maps) – це процес збереження деталізації розробленої високо-полігональної моделі на низько-полігональну модель у вигляді карт(базових текстур).

6.Текстурування (Texturing) – створення текстур, які будуть визначати кольори, матеріали та властивості моделі. Простими словами текстурування надає моделі індивідуальність та чіткість розуміння, що юзер бачить на екрані свого монітору. В рамках проєкту стилістика текстур визначається на першому етапі розробки, а саме – концептування.

8.Створення матеріалів(Material Creation) – процес налаштування матеріалів у 3D редакторі для надання моделі фінального вигляду.

9. Інтеграція в ігровий рушій (Integration into Game Engine) – процес імпорту моделі в ігровий рушій. Цей етап є фінальним.

Отже, проаналізувавши дану методику створення 3D моделі, були виявлені переваги та недоліки:

Переваги методу для створення 3D графіки для з тематикою виживання в скрутних умовах:

Ідеально підходить для проєкту створення графіки для гри на тематику виживання у скртуних умовах, тому що методика зумовлена на поетапному процесі розробки, а саме кожен етап не може бути початий поки не завершиться попередній. Обґрунтуванням ще служить те, що наша система управління базується на методі Waterfall.

Недоліки методу для створення 3D графіки для з тематикою виживання в скрутних умовах:

Цей процес вимагає багато часу та є дуже трудомістким. Технологічні обмеження, використання певних інструментів та програмного забезпечення може мати своє обмеження.

Наступне завдання, а саме другий об’єкт дослідження – 3D редактор. Метою буде проаналізувати та визначити в якому технічному середовищі та за допомогою яких інструментів здійснюється розробка гейм реді 3D моделі та пошук можливості інтеграції власного програмного продукту. У цьому пункті буде розглянуто основні технічні засоби та програмне забезпечення, яке використовується для створення геймреді 3D моделей.

Розробка геймреді 3D моделі вимагає використання спеціалізованого програмного забезпечення. Важливо обрати правильні інструменти та налаштувати середовище для ефективної роботи, що забезпечить високу продуктивність розробки моделей та їх оптимізацію для використання в реальному часі для проєкту.

Отже, для початку потрібно визначити що таке 3D редактор:

3D редактор — це програмне забезпечення, яке використовується для створення тривимірних (3D) моделей, сцен та анімацій. Ці інструменти дозволяють художникам, дизайнерам і розробникам створювати детальні об'єкти та оточення, які можуть бути використані в різних галузях, таких як відеоігри, кіно, архітектура, віртуальна реальність та інші візуальні медіа.

3D редактори надають розробникам широкий спектр інструментів та функцій для роботи з тривимірними об'єктами. Тому далі буде розглянуто основні функції 3D редакторів:

1.Моделювання:
Полігональне моделювання: Створення та редагування моделей за допомогою полігонів (вершин, ребер і граней).
Скульптура: Інструменти для створення високодеталізованих моделей шляхом "ліплення" з використанням різних кистей та інструментів.
NURBS/Spline(НУРБС та сплайни): Техніки моделювання, що використовують криві для створення гладких та точних поверхонь.

2. Текстурування та Матеріали:
UV розгортка: Процес розгортання 3D моделі на 2D площину для нанесення текстур.
Шейдери та матеріали: Налаштування вигляду поверхонь моделі, включаючи кольори, блиск, прозорість та інші властивості.

3.Рендеринг:
Фотореалістичний рендеринг: Створення високоякісних зображень, що виглядають як фотографії.
Реального часу: Використовується для відеоігор та віртуальної реальності, де швидкість рендерингу є критичною.

Тепер після проведення аналізу технічного середовища та його основного функціоналу потрібно розглянути та визначити самий кращий 3D редактор з яким буде відбуватися розробка програмного плагіну в рамках проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах.

Популярні 3D Редактори сьогодення:

1.Blender: Безкоштовне та відкрите ПЗ, яке пропонує повний набір інструментів для моделювання, текстурування, анімації та рендерингу.

2.Autodesk Maya: Один з найбільш популярних інструментів у індустрії, використовується для створення складних анімацій та моделей.

3.3ds Max: Також розроблений Autodesk, широко використовується в архітектурній візуалізації та геймдеві.

4.Cinema 4D: Відомий своєю зручністю у використанні та популярний серед дизайнерів графіки та реклами.

5.ZBrush: Спеціалізований на цифровій скульптурі та використовується для створення високодеталізованих моделей.

Критерій	**Blender**	**Maya**	**3ds Msx**	**Cinema 4D**	**ZBrush**
Ціна	Безкоштовно	Платна підписка	Платна підписка	Платна підписка	Платна одноразова покупка
Інтерфейс	Висока гнучкість, складний	Професійний, складний	Інтуїтивний, складний	Інтуїтивний	Унікальний, спеціалізований
Моделювання	Відмінне, широкі можливості	Відмінне, професійні інструменти	Відмінне, професійні інструменти	Хороше, зручне для початківців	Обмежене базове моделювання
Плагіни та розширення	Величезна кількість безкоштовних та платних	Величезна кількість платних	Величезна кількість платних	Величезна кількість платних	Величезна кількість платних
Використання	Широке, особливо для інді-розробників	Широке, провідний стандарт у кіно та ігровій індустрії	Широке, особливо в архітектурній візуалізації	Широке, особливо в графічному дизайні та анімації	Спеціалізоване, широко використовується для скульптингу

Оскільки проєкт являється ІТ та повинен мати етап **розробки програмного продукту** потрібно обрати таке програмне забезпечення в який буде можливість інтегрувати власну розробку. Тому проаналізувавши список можливого програмного забезпечення вибір падає на 3D редактор **Autodesk Maya** від розробника Autodesk.

Переваги цього редактору для створення 3D графіки для гри з тематикою виживання в скрутних умовах:

1.Розширювана архітектура:

- API (Application Programming Interface): Maya має потужний та добре документований API, який дозволяє розробникам взаємодіяти з усіма аспектами програми. Це забезпечує можливість створення плагінів, які можуть додавати нові функції, автоматизувати завдання або інтегрувати Maya з іншими інструментами та системами створення 3D графіки для гри з тематикою виживання в скрутних умовах.

- Скриптинг на Python і MEL: Maya підтримує два основні скриптові мови - Python та MEL (Maya Embedded Language). Python є популярною мовою програмування, яка дозволяє легко створювати скрипти для автоматизації задач та розробки інтерфейсу користувача. MEL, з іншого боку, є рідною мовою Maya, що дозволяє доступ до всіх внутрішніх команд та функцій**.**

2. Інструменти для розробки:

- Autodesk Maya Devkit: Autodesk надає розробникам проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах набір інструментів та бібліотек, які спрощують процес розробки плагінів. Devkit включає приклади коду, документацію та інструменти для компіляції та тестування плагінів.

- Visual Studio Integration: Для розробників на Windows, Maya пропонує інтеграцію з Microsoft Visual Studio, що забезпечує зручне середовище для написання, налагодження та тестування плагінів.

3.Висока продуктивність:

- Оптимізовані алгоритми: Плагіни, написані для Maya, можуть використовувати потужні алгоритми та інструменти для обробки великих об'ємів даних та складних 3D сцен. Це дозволяє забезпечити високу продуктивність та ефективність роботи.

- Підтримка багатоядерності та GPU: Maya використовує переваги сучасних процесорів та графічних карт, що дозволяє розробникам створювати плагіни, які використовують багатоядерні процесори та GPU для прискорення обчислень.

У підсумку Autodesk Maya має потужні можливості для розробки плагінів завдяки своїй розширюваній архітектурі, великій спільноті розробників, інтеграції з іншими програмами, підтримці стандартів обміну даними та наданню інструментів для розробки. Це робить Maya ідеальним вибором для створення індивідуальних рішень, які розширюють функціональність програми та забезпечують ефективність і високу якість роботи в індустрії 3D моделювання.

Отже, 3D редактор є невід'ємним інструментом для сучасних цифрових художників та розробників. Вони надають можливість створювати візуально привабливий та технічно оптимізований контент, який можна використовувати у різноманітних медіа та індустріях. В рамках дипломного проєкту, вивчення та використання 3D редактора дозволить створити високоякісні геймреді моделі та інтегрувати власний програмний продукт, що є основою для успішної розробки відеоігор та інших інтерактивних додатків.

Наступним кроком буде розглянута та обрана методологія створення плагінів для Autodesk Maya. Почнемо з визначення що таке плагін:

Плагін - це програмне забезпечення, яке додає певні функції до існуючої програми. Це модуль, який інтегрується з базовою програмою для розширення її можливостей без необхідності модифікації основного коду.

Тепер визначимо навіщо буде створений плагін для програмного забезпечення AutoDesk Maya в рамках дипломного проєктування. Отже, для поліпшення ефективності створення 3D графіки для гри з тематикою виживання у скрутних умовах знадобиться універсальний інструмент який зможе забезпечити розробникам створювати тривимірну графіку більш швидше, що буде сприяти зменшенню витрат у часі та бюджеті проєкту.

На сьогоднішній день існує два методи створення плагінів для Autodesk Maya:

Метод 1: Створення плагіну на основі Python.

Python — потужна мова програмування, яка використовується для створення плагінів у Autodesk Maya. Цей метод передбачає використання Python API для взаємодії з функціоналом Maya.

Переваги використання Python для проєкту створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

1.Зручність використання – Пайтон має простий та зрозумілий синтаксис, що дозволить швидко писати код.

2.Ширика підтримка – Пайтон широко використовується в індустрії та надає велику кількість ресурсів та бібліотек.

3.Модульність – можливість розробляти модульні та масштабовані програми завдяки об’єктно орієнтованому програмуванню.

4.Підтримка розширень – легко інтегрувати з іншими мовами, інструментами та програмним забезпеченням.

5.Динамічність – дозволяє швидко прототипувати і тестувати ідеї.

Недоліки використання Python для проєкту створення 3D графіки для комп’ютерної гри з тематикою виживання в скрутних умовах:

1.Швидкодія – пайтон може призводити до нижчої продуктивності порівняно з компільованими мовами.

2.Залежності – плагіни можуть вимагати додаткових бібліотек.

Метод 2: Створення плагіну на основі MEL.

MEL (Maya Embedded Language) — мова сценаріїв, спеціально розроблена для Autodesk Maya. MEL дозволяє автоматизувати різні завдання та створювати плагіни для Maya.

Переваги використання MEL для проєкту створення 3D графіки для комп’ютерної гри з тематикою виживання в скрутних умовах:

1.Інтеграція – ця мова тісно інтегрована з рідним програмним забезпеченням і надає прямий доступ до функціоналу.

2.Простота – гарно забезпечує прості автоматизації та швидкі сценарії.

3.Документація – широка документація та приклади.

Недоліки використання MEL для проєкту створення 3D графіки для комп’ютерної гри з тематикою виживання в скрутних умовах:

1.Обмеженість – обмежені можливості порівняно з Python.

2.Масштабованість – менш підходить для створення складних та великих плагінів.

3.Підтримка – обмежена підтримка об’єктно-орієнтованого програмування та сучасних концепцій розробки.

Порівняння Python та MEL:

Аспект	Python	MEL
Зручність	Зручний, зрозумілий синтаксис	Простий для базових автоматизацій
Підтримка	Широка підтримка, багато ресурсів	Широка документація, інтеграція з Maya
Можливості	Потужність, підтримка ООП, модульність	Обмежені, підходить для простих задач
Продуктивність	Нижча швидкодія через інтерпретовану природу	Вища швидкодія для простих задач
Масштабованість	Висока, підходить для складних плагінів	Обмежена, складно підтримувати великі проєкти
Розширюваність	Легке інтегрування з іншими мовами	Обмежена розширюваність

Враховуючи усі переваги та недоліки вибір падає на Python, оскільки він дійсно має зручніший функціонал, а саме головне, що на ринку вакансій знайти розробника Python набагато легше.

Отже, для проєкту управління створення 3D графіки для гри з тематикою виживання в скрутних умовах буде обрано перший метод для створення плагіну на мові Python.

2.4 Побудова інформаційної моделі

Опираючись на інформацію з першого розділу було обрано метод управління – Waterfall. Перейдемо до обґрунтування чому саме Waterfall:

1.Стабільність вимог: В нашому проєкті чітко визначений набір вимог і функціональних можливостей. Вимоги створені на початку проєкту не будуть піддаватися значним змінам протягом усього життєвого циклу проєкту, це забезпечує стабільну основу для розробки.

2.Послідовний характер завдань: В рамках проєкту усі етапи розробки передбачені послідовним методом виконання для забезпечення систематичного прогресу.

3.Фіксований бюджет та графік: проєкт має фіксований бюджет. Це забезпечує керування ресурсами та витратами на кожному етапі.

4.Чіткість управління: Кожен етап проєкту буде задокументований, це посприяє зменшенню ризиків.

5.Невеликі зміни в процесі розробки: проєкту не потрібно швидко реагувати на зміни вимог або виявлені помилки. Кожен перехід до наступного етапу буде завершуватись виконанням минулого.

В цьому пункті буде побудована інформаційна модель щодо реалізації інтеграції програмного продукту в програмне забезпечення з переходом до опису створення кінцевого продукту.

Отже, почнемо з огляду етапів розробки інтеграції в програмне забезпечення (плагіну) для інформаційної моделі:

1. Визначення вимог до плагіну
- Вхідні дані: Технічне завдання, специфікації, функціональні вимоги.
- Вихідні дані: Документ з вимогами до плагіну.
2. проєктування плагіну
- Вхідні дані: Документ з вимогами до плагіну.
- Вихідні дані: Архітектура плагіну.
3. Розробка плагіну
- Вхідні дані: Архітектура плагіну.
- Вихідні дані: Код плагіну, документація.
4. Тестування
- Вхідні дані: Код плагіну, документація.
- Вихідні дані: Тестові кейси, звіт про тестування.
5. Інтеграція плагіну
- Вхідні дані: Протестований код плагіну.
- Вихідні дані: Інтегрований плагін в Autodesk Maya, документація для розробника.

Далі будуть показані етапи розробки 3D моделей:

1.Концептуалізація та створення концепт артів
- Вхідні дані: Ідеї, референси, технічні вимоги.
- Вихідні дані: Концепт-арти.
2.3D моделювання
- Вхідні дані: Концепт-арти.
- Вихідні дані: Базові 3D моделі.
3.Створення матеріалів та текстур
- Вхідні дані: Базові 3D моделі.
- Вихідні дані: Текстури, матеріали.
4.Анімація

- Вхідні дані: 3D моделі з матеріалами.
- Вихідні дані: Скелетні структури.
5.Тестування та оптимізація
- Вхідні дані: 3D моделі, текстури, анімації.
- Вихідні дані: Оптимізовані моделі, звіт.
6. Інтеграція в ігрове середовище
- Вхідні дані: Оптимізовані моделі
- Вихідні дані: Готові для використання геймреді 3D моделі в ігровому рушії.

2.5 проєктування системи

У розділі проєктування системи буде наведені алгоритми розробки програмного забезпечення та алгоритм створення 3D графіки в межах проєкту.

Спочатку наведемо визначення що таке діаграма послідовностей:

Діаграма послідовностей - це діаграма, на якій показані взаємодії об'єктів, упорядковані за часом їхнього прояву. На діаграмі послідовності неявно присутня вісь часу, що дозволяє візуалізувати тимчасові відношення між переданими повідомленнями.

В даному випадку діаграма поділена на два блоки.

Перший блок: Розробка Плагіну (Рисунок 2.5.1)

В цьому блоці описана послідовність виконання стаком розробників процесу розробки плагіну для AutoDesk Maya.

A)Збір вимог до плагіну: На цьому етапі визначаються цілі, завдання та вимоги до плагіну, що дасть чітке уявлення кінцевого рузультату та його функціональні можливості. Збір вимог є критично важливим для розуміння необхідного функціоналу плагіну та запобігання невідповідностям у майбутньому.

B)Налаштування середовища розробки: Після визначення вимог необхідно підготувати середовище для розробки, включаючи встановлення необхідних бібліотек та підключення до API Autodesk Maya. Це забезпечує розробникам можливість безперешкодно працювати з необхідними інструментами та технологіями.

C)Розробка основних функцій плагіну: Основний функціонал плагіну розробляється на цьому етапі. Включає написання коду та первинну перевірку його працездатності. Це є основною частиною процесу розробки, де створюється ядро функцій плагіну.

D)Розробка графічного інтерфейсу (GUI): Для забезпечення зручності розробників, розробляється графічний інтерфейс. Створення інтуїтивно

зрозумілого та функціонального GUI є важливим для зручності використання плагіну кінцевими користувачами.

E)Тестування плагіну: Тестування дозволяє виявити та виправити помилки, забезпечити стабільну роботу плагіну та відповідність його початковим вимогам. Це важливий етап для досягнення високої якості продукту.

F)Документація плагіну: Документація описує функціональні можливості плагіну, спосіб його використання та вирішення можливих проблем. Це забезпечує користувачів необхідною інформацією для ефективного використання плагіну.

Створення геймреді 3D моделей: Після завершення роботи над плагіном, починається етап створення геймреді 3D моделей. Цей процес включає визначення технічних вимог, створення концепт-артів, моделювання, текстурування, анімацію та тестування моделей.

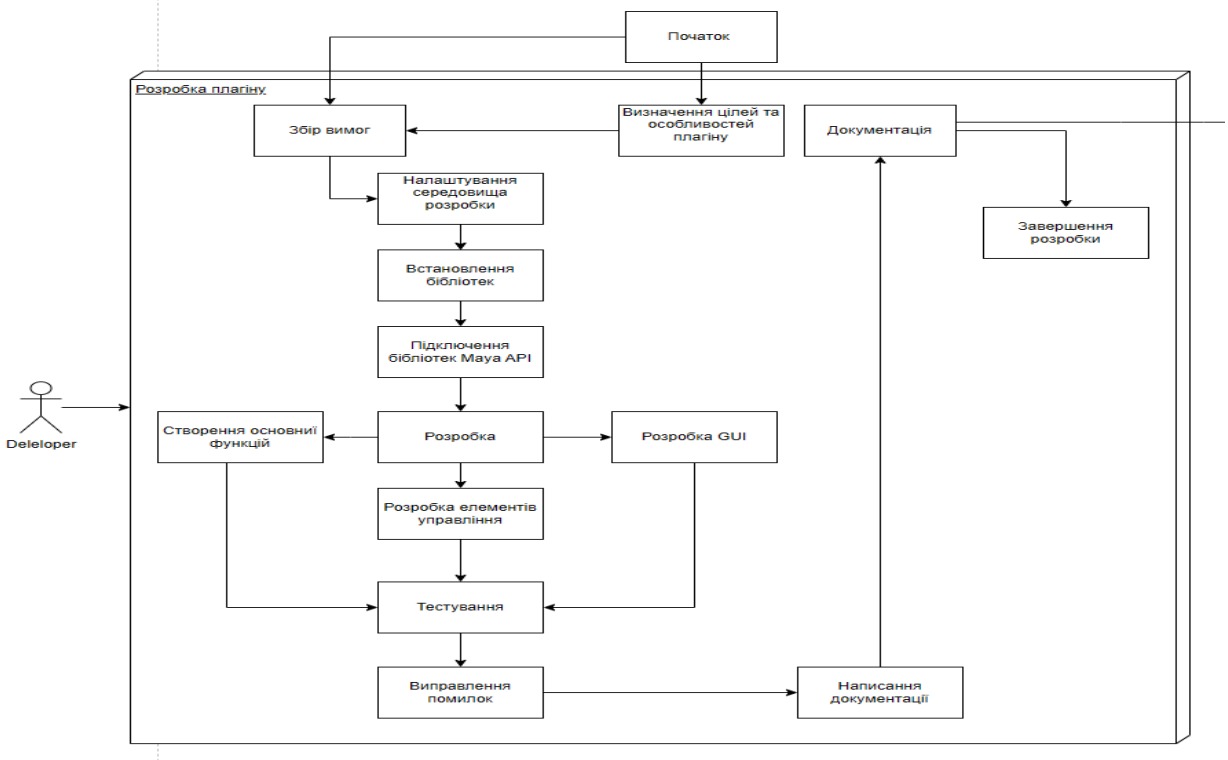


Рисунок 2.5.1 (Блок Розробки Плагіну)

Другий блок: Розробка 3D графіки (Рисунок 2.5.2)

В другому блоці зображений процес створення 3D графіки для гри на тематику виживання в скрутних умовах.

Важливо зазначити, що процес ‘‘Документація’’ з першого блоку розробки плагіну переходить до другого.

Документація переходить до стаку розробників тривимірної графіки і у результаті команда починає розробки з власним програмним забезпеченням, що значно підвищить продуктивність та зменшить витрати часу.

Усі етапи розробки 3D графіки детально описані у підрозділі ‘‘Виділення та аналіз предметної області’’.

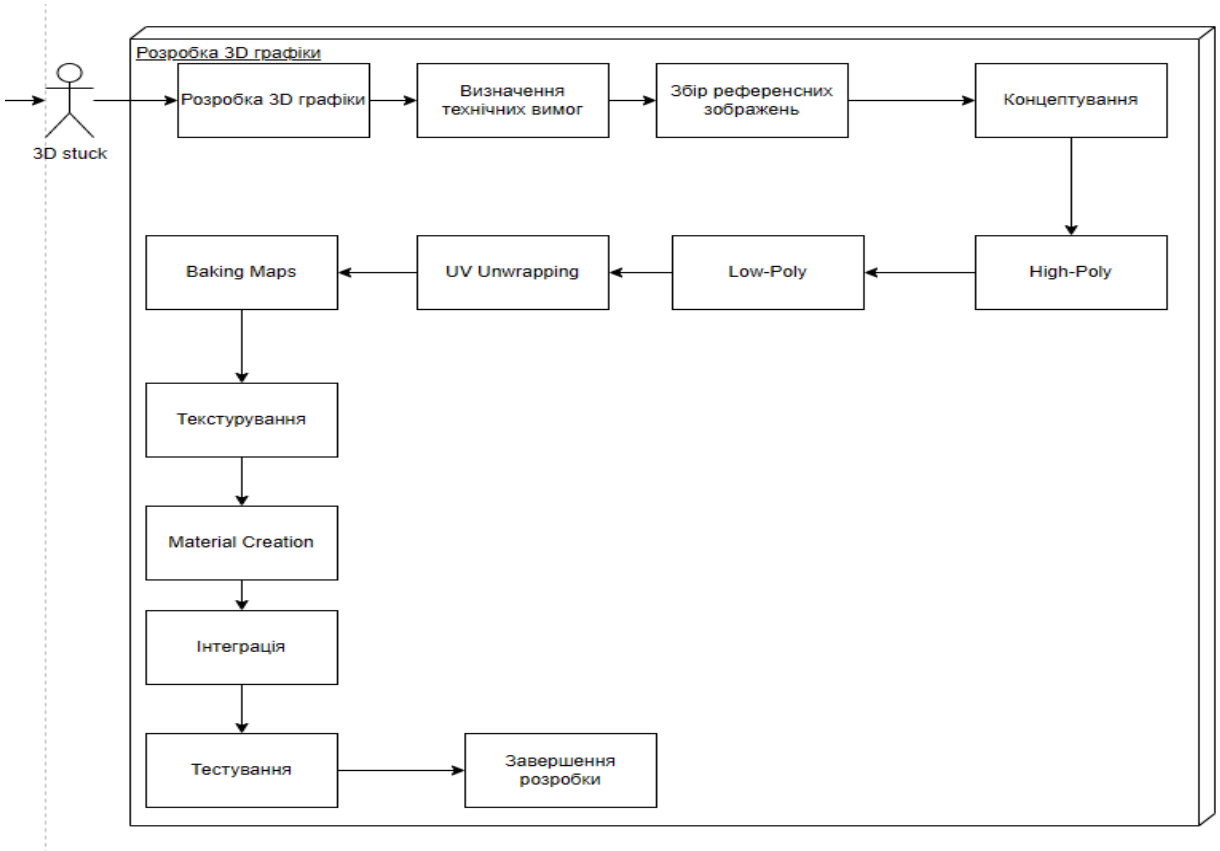


Рисунок 2.5.2 (Блок Розробки 3D Графіки)

Отже, дотримання цього алгоритму забезпечує структурованість та послідовність у процесі розробки, дозволяє мінімізувати ризики та підвищити якість кінцевого продукту. Обраний підхід забезпечує інтеграцію розроблених плагінів та створених моделей у робочі процеси, що сприяє ефективному досягненню поставлених цілей проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах.

2.6 Математичне та алгоритмічне забезпечення

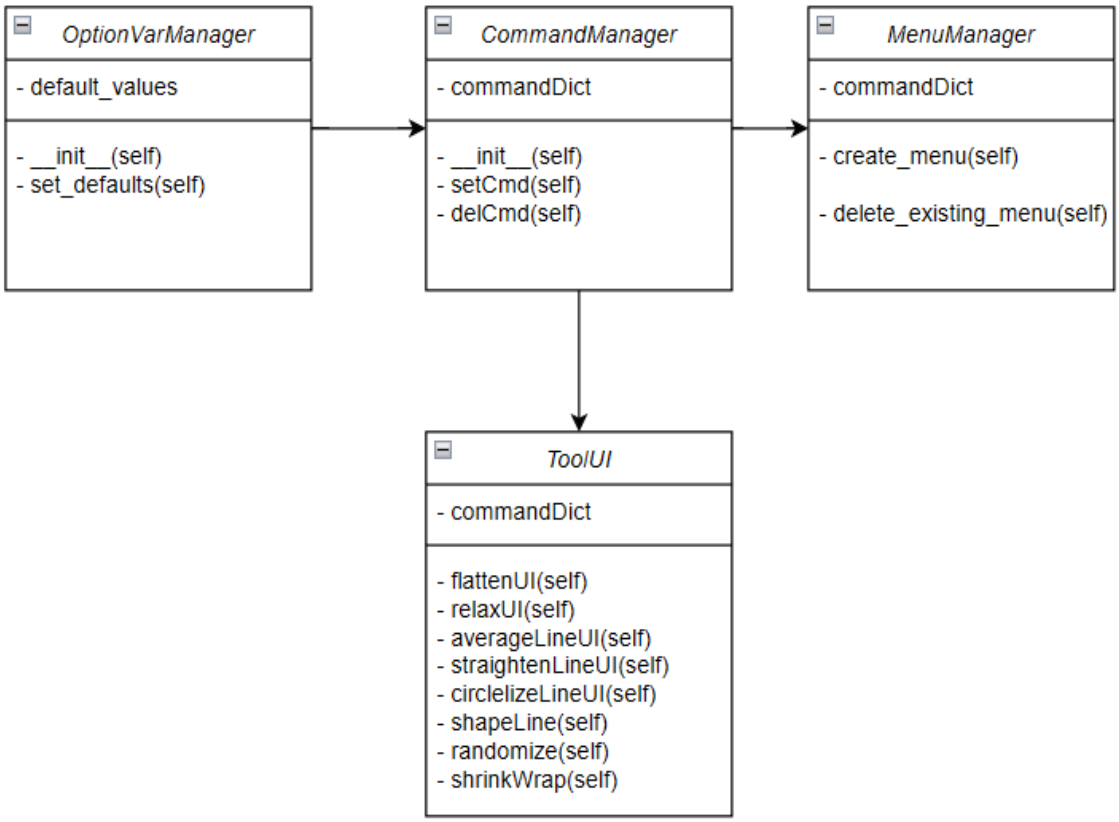
У підрозділі "Математичне та алгоритмічне забезпечення" представлено детальний аналіз математичних моделей, алгоритмів та методів, які використовуються для розробки плагіну для Autodesk Maya. Метою цього розділу

є обґрунтування вибору математичних методів та алгоритмів, що забезпечують ефективну і коректну роботу плагіну, а також високу якість кінцевих 3D моделей.

У цьому підрозділі буде виконано наступне завдання:

Опис діаграми реалізації методів: Будуть розглянуті алгоритми, які використовуються для розробки плагіну на Python, а також порівняння їх з алгоритмами на MEL. Обґрунтування вибору Python як основної мови програмування для реалізації плагіну.

UML-діаграма класів програмного продукту (Малюнок 3.2.1) для проєкту створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:



Малюнок 3.2.1 – UML діаграма класів.

Ця структура дозволяє легко розширювати та підтримувати код, оскільки кожен клас відповідає за окремий функціональний блок.

2.7 Висновки

1. Визначено мету дослідження предметної області.
2. Виділені та проаналізовані об'єкти дослідження.

3. Визначені недоліки та переваги об'єктів дослідження.
4. Проведено порівняння об'єктів дослідження.
5. Проаналізовані методи розробки.
6. Побудована інформаційна модель.
7. Розроблено проєктування системи розробки.
8. Проведено математичне та алгоритмічне забезпечення.
9. Створено UML-діаграму класів програмного коду.

РОЗДІЛ 3 проєктні рішення та розробка програмного продукту.

3.1. Технології реалізаціїї программного продукту.

Розробка програмного продукту вимагає ретельного планування та виконання декількох етапів, починаючи від концептуалізації та закінчуючи розгортанням готового продукту. Ця технологія описує процес створення 3D графіки для гри на тематику виживання в скрутниї умовах, включаючи розробку плагіну для Autodesk Maya на Python, створення геймреді 3D моделей та їх інтеграцію в ігрове середовище.

У цьому розділі буде наданий чіткий опис технологій реалізації програмного продукту. Також будуть наведені переваги та недоліки.

3.2 Вибір мови програмування

Мова Python – це автономна мова сценаріїв, вона не залежить напряму від системи Maya, дуже популярна мова в області створення 3D графіки. В нашому випадку, а саме створення трьовимірної графіки для гри, необхідно працювати з таким форматом як json, або з власними форматами даних, і Python має велику підтримку в цій області.

Плагін створений мовою програмування Python з використанням API Autodesk Maya. Python обраний через його простоту, читабельність і потужні можливості для автоматизації.

Python є популярною мовою серед художників і технічних фахівців у сфері комп'ютерної графіки завдяки своїй здатності швидко створювати прототипи і розробляти складні інструменти з мінімальним обсягом коду.

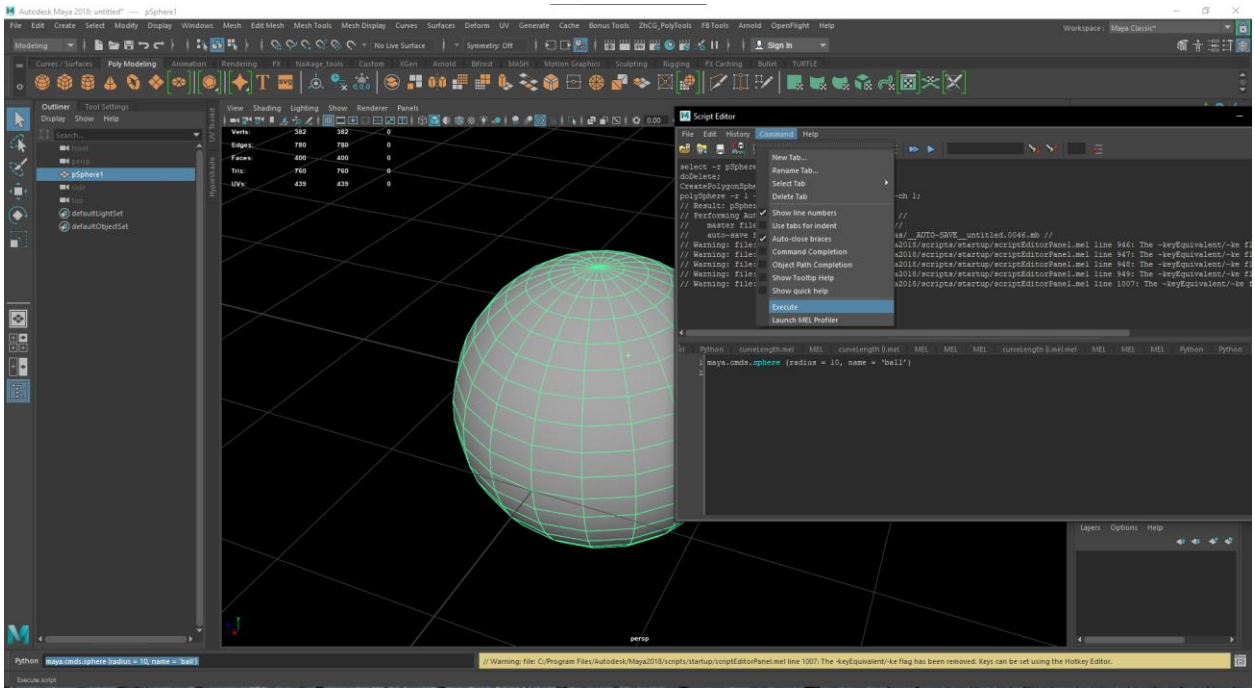
1.Приклад використання команди Python в Maya (Малюнок 3.2.1):

maya.cmds.sphere (radius = 10, name = ‘ball’)

Ця команда створює сферу в Maya.

Префікс maya.cmds вказує Python на те, що використовується команда з Maya Embedded Language (MEL). У цьому випадку, команда sphere отримує аргументи radius та name, які визначають радіус сфери та її назву відповідно. На відміну від звичайного синтаксису MEL, де ці параметри можуть передаватися як прапори, у Python вони передаються як аргументи функції.

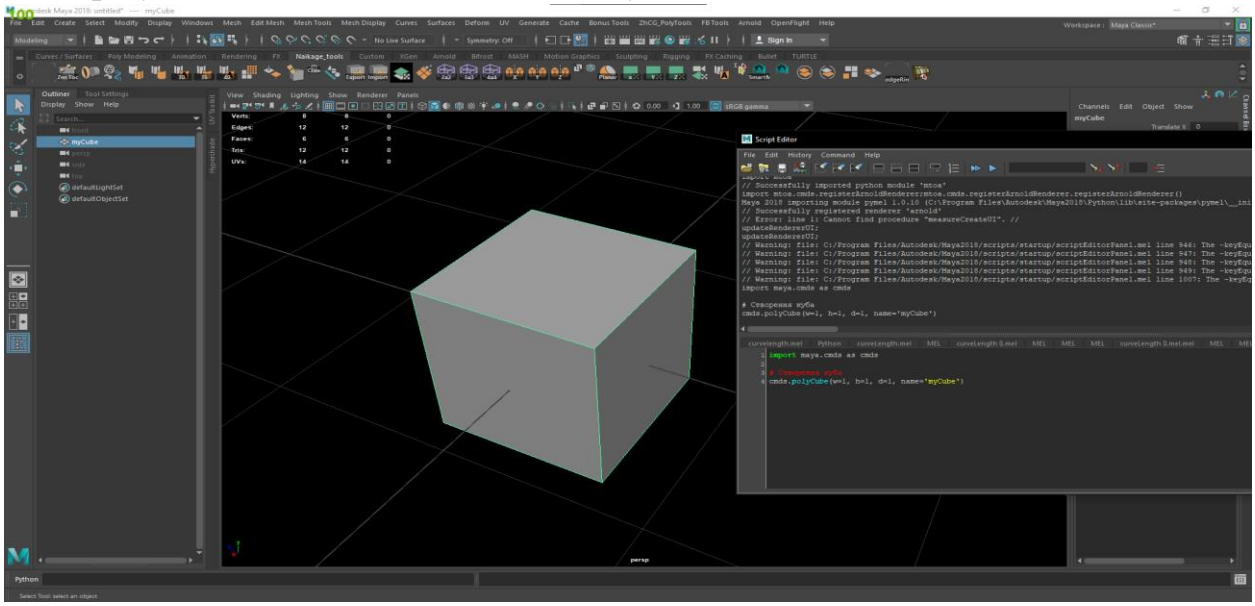
Таким чином, команда maya.cmds.sphere(radius=10, name='ball') створює сферу з радіусом 10 одиниць і назвою "ball", використовуючи Python API для Maya.



Малюнок 3.2.1 – Створення сфери за допомогою команди Python.

2.Приклад використання команди Python в Maya (Малюнок 3.2.2):

```
import maya.cmds as cmds

# Створення куба

cmds.polyCube(w=1, h=1, d=1, name='myCube')
```



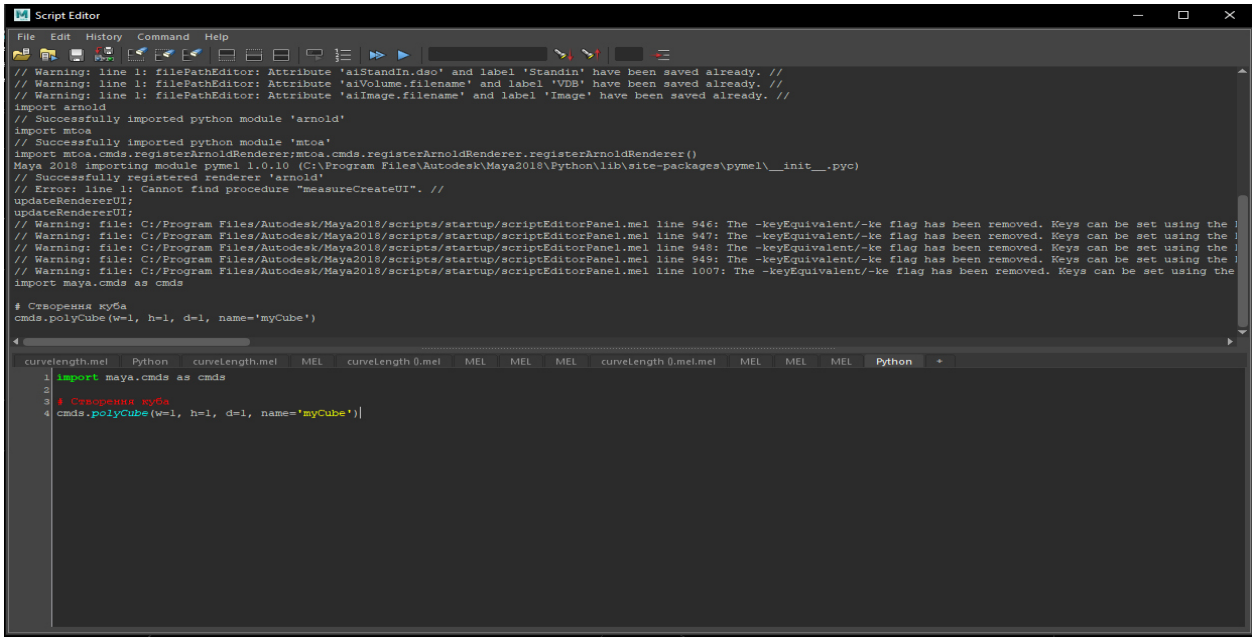
Малюнок 3.2.1 – Створення куба за допомогою команди Python.

Ця команда cmds.polyCube створює полігональний куб з вказаними параметрами w – ширина, h- висота, d- глибина.

Використовується модуль maya.cmds, який містить команди для роботи з об’єктами.

3.3 Вибір середовища програмування

Код розробляється в середовищі Autodesk Maya (Зображеня 3.3.1) - це інструменті для 3D моделювання. Дане програмне забезпечення має багатий набір інструментів для створення 3D графіки та API для інтеграції з Python, що дозволяє вдосконалювати функціональність розробникам та одночасно реалізовувати цілі проєкту створення 3D графіки для з тематикою виживання в скрутних умовах.



Малюнок 3.3.1 – Середовище програмування Maya.

3.4 Використання технологій

Для створення плагіну використовується API Autodesk Maya.

API Autodesk Maya надає можливість керувати 3D об'єктами, створювати нові моделі, застосовувати матеріали, налаштовувати освітлення, анімувати об'єкти і багато іншого. Основні модулі, що використовуються в коді, включають maya.cmds і maya.mel.

maya.cmds: Цей модуль надає доступ до команд Maya через Python. Він дозволяє створювати і маніпулювати 3D об'єктами, а також налаштовувати параметри сцен.

maya.mel: Модуль дозволяє виконувати MEL (Maya Embedded Language) команди з Python, що забезпечує зворотну сумісність з існуючими MEL скриптами.

Технологія ‘optionVar’

В розробці програмного продукту даного дипломного проєктування використовується технологія optionVar для збереження налаштувань розробника між сеансами роботи з Maya.

Ця технологія надає можливість зберігати цілі числа, рядки і значення з плаваючою точкою, забезпечуючи гнучкість у збереженні різних типів даних.

Отже, тепер потрібно визначити переваги використання Python для плагінів Maya.

3.5 Виявлення переваг та недоліків використання Python та API Maya.

Переваги використання Python в даному дипломному проєктувані.

1.Перевага простоти та читабельності в проєкті створення 3D графіки для з тематикою виживання в скрутних умовах:

Пайтон має гарну властивість простотою написання коду та легкою читабельністю. Його синтаксис зрозумілий, що у результаті робить код простим для удосконалення та підтримки для нових версій.

2.Перевага наявності великої кількості бібліотек для проєкту створення 3D графіки для з тематикою виживання в скрутних умовах:

Пайтон має в наявності та включає у себе велику кількість бібліотек, це дозволяє розробнику програмного забезпечення швидко розшити функціональність та використовувати вже готові рішення для виконання різних простих задач.

3.Переваги використання інтеграцій з Maya в проєкті створення 3D графіки для з тематикою виживання в скрутних умовах:

Використання API Maya дозволяє інтегрувати розробнику плагін написаний на мові програмування Python безпосередньо в середовище створення 3D графіки, це надає змогу автоматизувати багато різних процесів та розширювати можливості програмного забезпечення.

4.Гнучкість використання мови Python в проєкті створення 3D графіки для з тематикою виживання в скрутних умовах:

Мова програмування Python забезпечить розробнику даного дипломного проєктування легко розробляти складні інструменти та прототипи з не тяжкими зусиллями. Завдяки гнучкості пайтон має підтримку різних парадигм методів програмування, включаючи у себе об’єктно-орієнтоване, функціональне та процедурне програмування.

Недоліки використання Python в даному дипломному проєктуванні.

1.Перевага продуктивності мови програмування Python в рамках розробки 3D графіки для гри з тематикою виживання в скрутних умовах:

Оскільки мова Python не є компільованою мовою програмування, це може призводити до нижчої продуктивності створення порівняно з компільованими мовами, такими як C++ та Java. В рамках дипломного проєктування це може бути критичним, якщо знадобиться обробка великих обсягів даних.

2.Переваги динаміки мови Python в розробці програмного продукту для проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Мова Python є мовою з динамічною типізацією, це може призвести до помилок які буде важко виявити на етапах розробки програмного продукту. В результаті можуть виявлятися такі помилки як збільшення витрати часу на тестування та відладку коду.

Переваги та недоліки використання Maya API:

Переваги використання Maya API в проєкті створення 3D графіки для комп'ютерної гри на тематику виживання в скрутних умовах:

1.Перевага наявності великого набору інструментів для використанні у проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Використання середовища Maya API має доступ до великої кількості інструментів 3D моделювання. Це дозволяє створювати складні та інтерактивні інструменти для роботи з 3D графікою в рамках дипломної роботи для гри на тематику виживання в скрутних умовах.

2.Переваги використання інтеграцій з MEL в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

API Maya має зв'язок з MEL(Maya Embedded Language), це зумовлено тим, що при розробці програмного продукту є можливість використовувати старі інструменти.

Недоліки використання Maya API в проєкті створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

1.Наявність складності використання в даному дипломному проєктуванні:

Для роботи з Maya API вимагається наявність знань про внутрішні механізми та інструменти Maya та принципи роботи з 3D графікою.

2.Наявність дефіциту продуктивності використання Maya API в створенні 3D графіки для гри з тематикою виживання в скрутних умовах:

Компілювання складних операцій через API може бути повільним, особливо якщо у сцені присутня велика кількість об’єктів. Це може вимагати додаткової оптимізації коду.

Використання Python у поєднанні з API Maya для розробки скриптів і інструментів має сильний інструмент для автоматизації і розширення можливостей Maya.

Проаналізувавши переваги та недоліки, склався висновок, що Python є відмінним вибором для розробки в середовищі Maya для створення 3D графіки у грі на тематику виживання в скрутних умовах.

3.6 Опис технології збереження даних.

У процесі розробки інструментів для 3D моделювання в дипломному проєктуванні важливо мати можливість зберігати користувацькі налаштування між сеансами роботи.

У програмному забезпеченні Autodesk Maya для цього використовується спеціальний механізм, який називається optionVar. Цей механізм дозволить зберігати різні типи даних і забезпечить їх доступність при наступних запусках програми.

У цьому підрозділі буде розглянуто основні аспекти технології збереження даних за допомогою optionVar, її переваги та недоліки.

Технологія збереження даних у Maya за допомогою optionVar.

Функція optionVar в проєкті використовується для збереження і отримання значень змінних, які можуть бути або числовими (iv для цілочислових значень), або строковими (sv для строкових значень), або з плаваючою точкою (fv для значень з плаваючою точкою). Ці змінні використані для зберігання налаштувань розробника, які залишатимуться доступними навіть після перезапуску Maya.

Ось представлені приклади використання optionVar в розробленому програмному продукті:

1.Приклад використання optionVar в проєкті розробки 3D графіки для з тематикою виживання в скрутних умовах:

```
if not mc.optionVar(ex=`GAME_POLY_TOOLS_polyTools _flatten_axis'):
    mc.optionVar(sv=[`GAME_POLY_TOOLS_polyTools _flatten_axis', 'view plane'])
```

У цьому прикладі перевіряється, чи існує змінна `GAME_POLY_TOOLS_polyTools _flatten_axis`.

Якщо змінна не існує, створюється нова змінна зі значенням 'view plane'. Таким чином, налаштування зберігається і може бути використане пізніше у скрипті або в іншому сеансі Maya.

2.Приклад використання optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Налаштування осі для вирівнювання(Flatten Axis):

```
if not mc.optionVar(ex='GAME_POLY_TOOLS_flatten_axis'):
   mc.optionVar(sv=['GAME_POLY_TOOLS_flatten_axis', 'view plane'])
```

Ця функція перевіряє чи існує змінна``GAME_POLY_TOOLS_polyTools_flatten_axis``. Якщо вона не існує, то плагін починає створювати нову змінну зі значенням ``view plane`` - це налаштування визначає по якій осі буде виконуватися вирівнювання.

Вирівнювання є важливим інструментом для роботи з трьовимірними моделями. Налаштування flatten_axis визначає, по якій осі буде здійснюватися вирівнювання, а flatten_space визначає, чи буде вирівнювання виконуватися у світовому чи об’єктному просторі.

3.Приклад використання optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Налаштування простору для вирівнювання(Flatten Space):

```
if not mc.optionVar(ex=' GAME_POLY_TOOLS _polyTools_flatten_space'):
   mc.optionVar(sv=[' GAME_POLY_TOOLS _polyTools_flatten_space', 'os'])
```

Цей приклад встановлює значення змінної GAME_POLY_TOOLS_flatten_space, якщо вона не існує. Значення 'os' означає, що вирівнювання буде виконуватися в об'єктному просторі.

4.Приклад використання optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Налаштування випадкових змін параметрів(Randomize Space).

```
if not mc.optionVar(ex=' GAME_POLY_TOOLS _randomize_space'):
   mc.optionVar(iv=[' GAME_POLY_TOOLS _randomize_space', 1])
```

В цьому випадку встановлюється значення змінної GAME_POLY_TOOLS_randomize_space, якщо вона не існує.

Значення ‘1’ вказує на використання об’єктного простору для рандомних змін параметрів.

Випадкові зміни використовуються для створення нерегулярних природних виглядів моделі. Налаштування randomize_space s randomize_shift дозволяють контролювати масштаб та простір для випадкових змін. В даному проєкті це буде корисно для розробки органічних форм та поверхонь.

5.Приклад використання optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Налаштування для збереження границь під час використання функції Relax.

```
if not mc.optionVar(ex='GAME_POLY_TOOLS_relax_keepBorder'):
    mc.optionVar(iv=['GAME_POLY_TOOLS_relax_keepBorder', 1])
```

Плагін зберігає налаштування GAME_POLY_TOOLS_relax_keepBorder, якщо воно не існує. Значення 1 показує, що границі будуть збережені під час виконання Relax.

Функція Relax використовується для згладжування моделей, це дозволить отримати більш природній вигляд. Налаштування relax_keepBorder визначає чи буде збережені границі моделі під час Relax – це важливо для збереження початкової форми.

3.7 Переваги та недоліки технології збереження даних за допомогою optionVar.

Переваги використання optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

1.Простота використання optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Ця функція забезпечує простий спосіб зберігати налаштування розробника без необхідності створення складних скриптів збереження. Всього декілька рядків коду мають можливість отримання та зберігання змінних.

2.Збереження між сеансами розробки використовуючи optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Данна функція дає можливість зберігати вказані значення розробника між сеансами роботи з програмним забезпеченням. Це означає, що налаштування розробника залишаються доступними після закриття і повторного запуску програми. Така особливість є корисною для збереження налаштувань інтерфейсу та інших.

3.Перевага гнучкості optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах :

Функція має можливість зберігати досить різні типи даних, такі як:

- Числа
- Рядки
- Рядки зі значеннями

Це забезпечує гнучкість у зберіганні різних налаштувань.

4.Можливість інтеграції з програмним забезпеченням в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Технологія optionVar інтегрується з Maya, це дозволяє спрощувати взаємодію з іншими інструментами та скриптами. Розробник може зосередиться на функціональності своїх інструментів, не переймаючись про сумісність механізмів збереження даних.

Недоліки використання optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

1.Обмеженість функціоналу використання optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Технологія optionVar підходить для простих налаштувань, але для складніших задач може знадобитися більш потужний механізм збереження.

2.Недолік продуктивності використання optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Велика кількість збережених змінних може вплинути на час зберігання та продуктивність системи.

3.Відстутність структурованості optionVar в проєкті створення 3D графіки для гри з тематикою виживання в скрутних умовах:

Функція не підтримує структуровані дані, це ускладнює процес керування великою кількістю інформації.

Технологія збереження даних за допомогою optionVar у Maya є ефективним інструментом для збереження користувацьких налаштувань між сеансами. Вона проста у використанні, гнучка та інтегрована безпосередньо з Maya, що робить її зручною для багатьох задач. Проте, для складніших задач може знадобитися використання інших технологій збереження даних. Враховуючи всі переваги та

недоліки, optionVar є відмінним вибором для збереження простих налаштувань, забезпечуючи при цьому безперервність і зручність роботи користувачів.

3.8 Опис додаткових технологій та підходів роботи программного забезпечення.

В цьому підрозділі буде надано детальне пояснення функцій програмного продукту (плагіну).

1. Ініціалізація налаштувань за допомогою optionVar.

Робота плагіну розпочинається з перевірки та встановлення значень змінних `optionVar`, якщо вони не існують.

```
if not mc.optionVar(ex='GAME_POLY_TOOLS_averageLine_stepMode'):
    mc.optionVar(iv=['GAME_POLY_TOOLS_averageLine_stepMode', 0])
if not mc.optionVar(ex='GAME_POLY_TOOLS_flatten_axis'):
    mc.optionVar(sv=['GAME_POLY_TOOLS_flatten_axis', 'view plane'])
if not mc.optionVar(ex='GAME_POLY_TOOLS_flatten_space'):
    mc.optionVar(sv=['GAME_POLY_TOOLS_flatten_space', 'os'])
if not mc.optionVar(ex='GAME_POLY_TOOLS_flatten_align'):
    mc.optionVar(sv=['GAME_POLY_TOOLS_flatten_align', 'mean'])
if not mc.optionVar(ex='GAME_POLY_TOOLS_relax_keepBorder'):
    mc.optionVar(iv=['GAME_POLY_TOOLS_relax_keepBorder', 1])
```

Цей фрагмент плагіну перевіряє наявність таких змінних:

- *Game_Poly_Tools_averageLine_stepMode*
- *Game_Poly_Tools_flatten_axis*
- *Game_Poly_Tools_flatten_space*
- *Game_Poly_Tools_flatten_align*
- *Game_Poly_Tools_relax_keepBorder*

Якщо вони не існують, створюються нові з відповідними значеннями.

2.Функція `setCmd()`.

Ця функція встановлює час виконання команд `runtimeCommand` у Maya.

```
def setCmd():
```

```
    for namedCommand, command in commandDict.items():
        try:
            mc.runTimeCommand(namedCommand, c=command, cat='Game_Poly_Tools', cl='python')
        except:
            pass
```

Функція перебирає словник `commandDict` і встановлює кожну команду як команду часу виконання. Якщо встановлення команди викликає помилку, вона просто пропустить її за допомогою конструкції `try…except`.

3.Функція `create_menu()`.

Данна функція створює користувацьке меню `Game_Poly_Tools` у головному вікні.

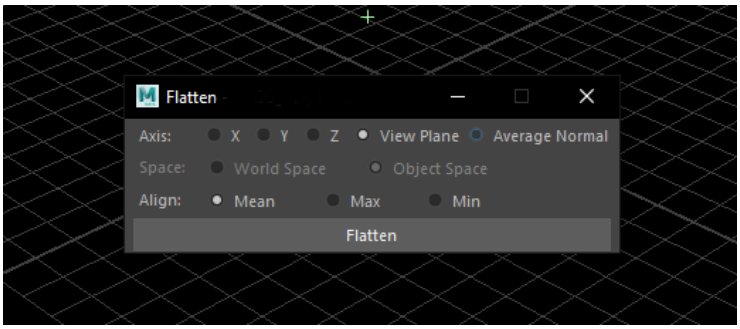
```
def create_menu():
    menus = mc.lsUI(menus=True)
    for eachMenu in menus:
        if 'Game_Poly_Tools' in mc.menu(eachMenu, q=True, l=True):
            mc.deleteUI(eachMenu)
            break

    melMenuCmds1 = '\n    menu -l "Game_Poly_Tools" -tearOff 1 -allowOptionBoxes 1 -p "MayaWindow"; …  n    menuItem -l "Harden Edges" -c "Display_HardenEdges";\n menuItem -l "Soften Edges" -c "Display_SoftenEdges";\n\n    setParent -menu ..;\n menuItem -d 1;\n    menuItem -l "v1.62" -en 0;\n    '
    melMenuCmds = melMenuCmds1
    mm.eval(melMenuCmds)
create_menu()
```

Функція перевіряє наявність меню і видаляє його, якщо воно існує, щоб уникнути дублювання. Далі йде створення нового меню з набором пунктів.

4.Функції інтерфейсу:

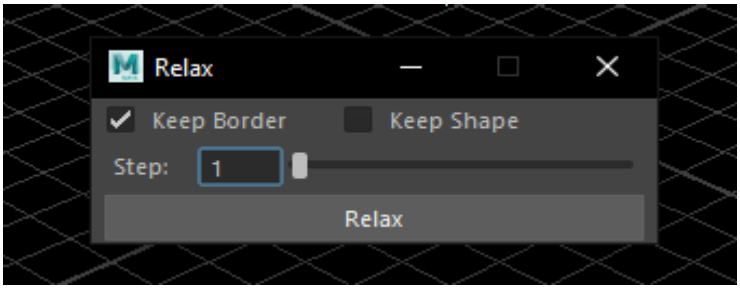
1.flattenUI() (Малюнок 3.8.1) - спрощує інтерфейс користувача, видаляючи або приховуючи непотрібні елементи, щоб зробити його більш зручним та

зрозумілим. Це може включати об'єднання схожих функцій, зменшення кількості відображуваних опцій або оптимізацію розташування елементів.



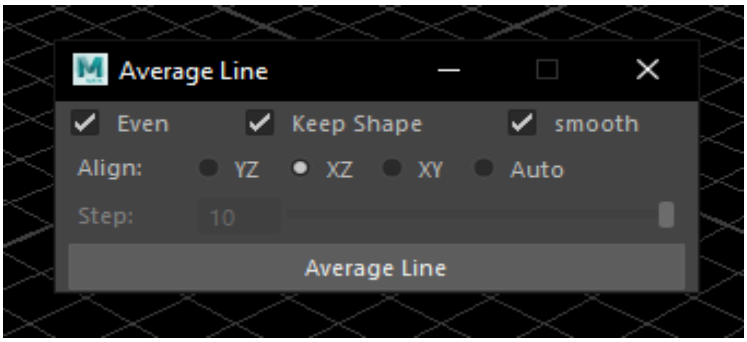
Малюнок 3.8.1 - flattenUI

2.relaxUI()(Малюнок 3.8.2) - згладжує та вирівнює вершини в 3D-моделі, зменшуючи деформації та нерівності, щоб поверхня виглядала більш гладкою.



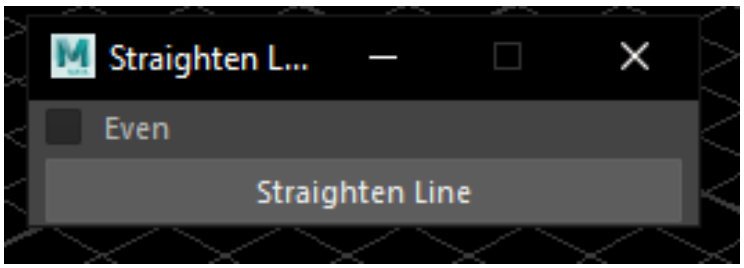
Малюнок 3.8.2 - relaxUI

3.averageLineUI()(Малюнок 3.8.3) - усереднює положення лінії між кількома заданими точками, вирівнюючи її для створення більш гладкого та рівного результату.



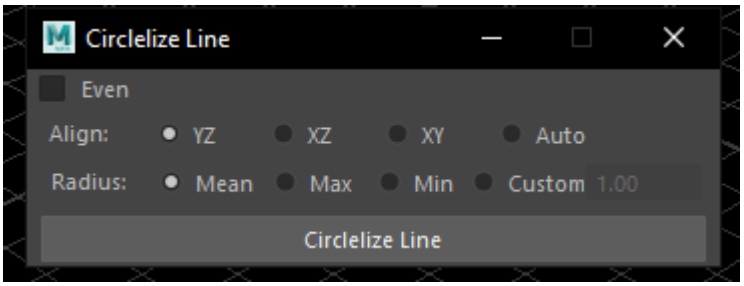
Малюнок 3.8.3 - averageLineUI

4.straightenLineUI()(Малюнок 3.8.4) - випрямляє криву або лінію, вирівнюючи її між двома або більше заданими точками для створення прямої лінії.



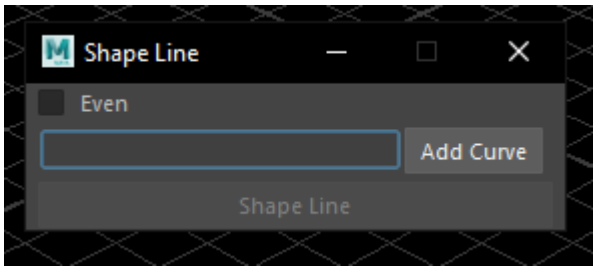
Малюнок 3.8.4 - straightenLineUI

5.circlelizeLineUI() (Малюнок 3.8.5)- перетворює лінію на форму, наближену до кола, розташовуючи її точки так, щоб вони утворювали колоподібну структуру.



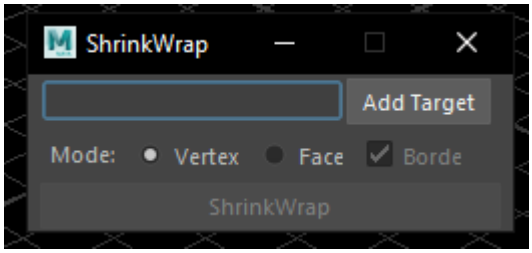
Малюнок 3.8.5 - circlelizeLineUI

6.shapeLineUI()(Малюнок 3.8.6) - змінює форму лінії, дозволяючи розташувати її точки відповідно до заданого шаблону або форми, створюючи певний контур або профіль.



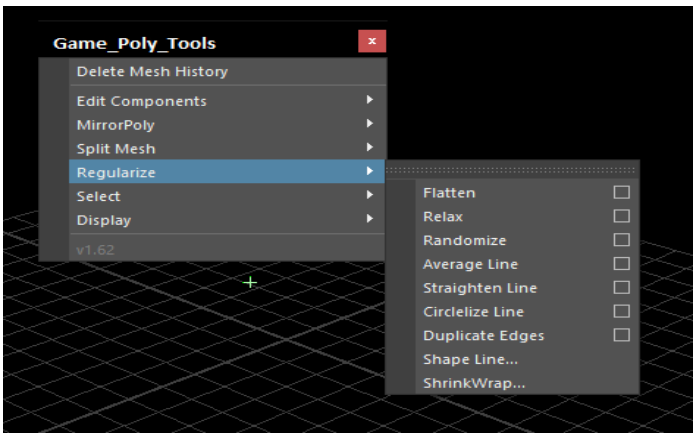
Малюнок 3.8.6 - shapeLineUI

7.shrinkWrapUI() (Малюнок 3.8.7)- обертає один об'єкт навколо поверхні іншого, притискаючи його до форми цільового об'єкта, подібно до обгортання.



Малюнок 3.8.7 - shrinkWrapUI

8.Основний інтерфейс (Малюнок 3.8.8).



Малюнок 3.8.8 – Основний інтерфейс.

Плагін Game_Poly_Tools використовує різноманітні функції та методи для взаємодії з Maya, налаштування інтерфейсу розробника та виконання операцій над 3D моделями. Широко використовується optionVar для збереження налаштувань, а також модулі maya.cmds і maya.mel для виконання команд Maya. Детальний аналіз цих функцій надав уявлення, як налаштовуються і виконуються різні операції в Maya.

У підсумку для даного дипломного проєктування було створено програмний продукт, який зебезпечить ефективну розробку 3D графіки для гри на тематику виживання в скрутних умовах.

3.9 Висновки

1. Визначені технології реалізації програмного продукту.
2. Обрана мова програмування.
3. Наведені приклади використання мови програмування в дипломному проєктуванні.
4. Обрано середовище програмування.
5. Описано використання технологій в дипломному проєктуванні.
6. Виявлені переваги та недоліки використання мов програмування.
7. Описані технології збереження даних.
8. Виявлені переваги та недоліки технологій збереження даних.
9. Описані додаткові технології та підходи роботи програмного забезпечення.

РОЗДІЛ 4 Управління проєктом

4.1 Загальний огляд проєкту

Для загального огляду цього розділу необхідно провести деякі уточнення щодо загального поняття що таке керування проєктом.

Управління проєктом - це систематичний підхід до планування, виконання, контролю та керування різними аспектами проєкту з метою досягнення його цілей в межах обмежень, таких як обсяг, час, бюджет та якість. Це процес, що включає управлінські зусилля, методи, інструменти та техніки для забезпечення ефективного виконання проєктних завдань та досягнення його результатів у визначені терміни та обсяги.

Управління проєктом передбачає планування ресурсів, взаємодію з учасниками проєкту, розподіл завдань, контроль прогресу та вчасну реагування на зміни. Його основна мета - забезпечити успішне завершення проєкту відповідно до вимог та очікувань зацікавлених сторін.

Оскільки проєкт зосереджений в IT-галузі, осі деякі аспекти управління IT проєктом для створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

1. Планування проєкту створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

Визначення цілей та обсягу робіт: Це включає чітке формулювання цілей проєкту, обсягу робіт, визначення вимог та специфікацій.

Розробка плану проєкту: Створення детального плану дій, що включає розклад завдань, розподіл ресурсів, визначення відповідальних осіб та встановлення контрольних точок.

2. Управління часом в проєкті створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

Складання графіка робіт: Розробка графіка з урахуванням всіх етапів проєкту, встановлення дедлайнів для кожного завдання.

Контроль термінів виконання: Моніторинг виконання завдань згідно графіку, виявлення затримок та своєчасне внесення коректив.

3. Управління ресурсами в проєкті створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

Розподіл ресурсів: Визначення та розподіл необхідних ресурсів (людських, технічних, фінансових) для виконання проєкту.

Оптимізація використання ресурсів: Забезпечення ефективного використання ресурсів для максимального досягнення результатів при мінімальних витратах.

4. Управління ризиками в проєкті створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

Ідентифікація ризиків: Виявлення потенційних ризиків, які можуть вплинути на виконання проєкту.

Аналіз і планування ризиків: Оцінка ймовірності виникнення та потенційного впливу ризиків, розробка планів щодо їх мінімізації або усунення.

5. Управління якістю в проєкті створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

Встановлення стандартів якості: Визначення вимог до якості продукту або послуги, які мають бути досягнуті в результаті проєкту.

Контроль якості: Постійний моніторинг та оцінка якості виконуваних робіт і кінцевого продукту, коригування процесів для досягнення необхідних стандартів.

6. Управління комунікаціями в проєкті створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

Планування комунікацій: Визначення методів та засобів комунікації між учасниками проєкту.

Забезпечення ефективної комунікації: Постійний обмін інформацією між членами команди, регулярні зустрічі, звітність перед зацікавленими сторонами.

7. Управління зацікавленими сторонами в проєкті створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

Ідентифікація зацікавлених сторін: Визначення всіх учасників, які впливають або зазнають впливу від проєкту.

Залучення та управління очікуваннями: Активна робота з зацікавленими сторонами для врахування їхніх вимог, очікувань та інтересів.

8. Управління змінами в проєкті створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах:

Контроль змін: Встановлення процесу управління змінами для забезпечення контролю над усіма змінами в проєкті.

Оцінка та впровадження змін: Оцінка впливу запропонованих змін, прийняття рішень про їх впровадження та інтеграція змін в план проєкту.

9.Завершення проєкту створення 3D графіки для комп’ютерної гри з тематикою виживання в скрутних умовах:

Оцінка результатів: Підсумкова оцінка досягнення цілей проєкту, аналіз відхилень від плану.

Документування уроків: Збір та документування уроків, отриманих під час виконання проєкту, для подальшого використання в майбутніх проєктах.

Закриття проєкту: Офіційне завершення проєкту, звільнення ресурсів, підготовка фінальних звітів та передача результатів замовнику.

Отже, основною ціллю управління IT-проєктом є досягнення усіх поставлених завдань та дотримання усіх поставлених обмежень та бюджету.

Ефективне управління IT-проєктом вимагає уваги до всіх цих аспектів та здатності гнучко реагувати на зміни і виклики, що виникають в процесі реалізації проєкту.

Ініціація проєкту розробки 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах стало спостереження за трендами в індустрії комп'ютерних ігор. Постійний попит на ігри з елементами виживання, які вміщують у себе глибокий сюжет та вражаючі візуальні ефекти. Такі проєкти привертають увагу ком’юніті своєю атмосферністю та здатністю поглинутись у світ екстремальних умов. Технології в галузі розробки 3D графіки створило нові можливості для відтворення реалістичних візуальних ефектів. Використання 3D графіки дозволяє поглибити гравців та забезпечити новий рівень взаємодії з грою.

4.2 Огляд задач проєкту

В цьому пункті буде розглянуто та показано основні задачі поставлені для досягнення розробки якісного продукту, а саме 3D моделей для гри.

Отже почнемо з пункту ‘‘Дослідження та ініціація’’(зображення 4.2.1).

На зображені можна побачити перелік завдань, ці завдання були створені для якісного старту проєкту.

1.Обсяг проєкту — ключовий аспект управління проєктом, в якому буде визначено його межі, цілі, завдання та кінцеві результати. Визначення обсягу проєкту є критичним для успішного виконання та завершення проєкту, оскільки він встановлює основу для планування, розподілу ресурсів, управління часом та контролю якості.

2. Бюджет проєкту — це детальний план, в якому визначено , скільки фінансових ресурсів буде потрібно проєкту для виконання всіх завдань та досягнення цілей. Він є основним фінансовим інструментом, який дозволяє керівнику проєкту

ефективно планувати, контролювати та управляти фінансами протягом усього життєвого циклу проєкту.

3. Project Overview — це загальний огляд проєкту, який включає ключову інформацію про його мету, завдання, обсяг, часові рамки, бюджет та основні етапи. Це короткий документ, який надає чітке уявлення про сутність проєкту, його значущість та основні моменти, що дозволяють всім зацікавленим сторонам швидко зрозуміти, про що йдеться в проєкті.

4. Ідентифікація стейкхолдерів — це процес в якому буде визначено усіх зацікавлених сторін, які можуть впливати на проєкт створення трьовимірної графіки для гри з тематикою виживання в скрутних умовах або підпадати під вплив його результатів. Це важливий етап управління проєктом, оскільки від правильного визначення та управління стейкхолдерами залежить успішне виконання проєкту.

5. Підготовка та оцінка беклогу проєкту — це процес в якому буде створено та підтримано список завдань, вимог та функцій, які необхідно виконати в рамках проєкту. Беклог є основним інструментом для управління та пріоритезації роботи нашої команди.

6.Підготовка складу команди – це етап у плануванні проєкту, він включає в себе визначенні необхідних кадрів та фахівців та забезпечення ефективного залучення до проєкту.

7.Підготовка JIRA flow - це процес в якому буде налаштовано організацію робочих процесів в системі управління проєктом Jira для забезпечення ефективного менеджменту завдань.

8.Підготовка Definition of Done - це процес в якому будуть визначені та уточнені критерії, які будуть визначати, коли робота над певним елементом проєкту може вважатися завершеною і готовою до випуску.

	Режим задачи	Название задачи	Длительнс	Начало	Окончани	Предшественн
0		**Проєкт**	**391 дней**	**Пн 29.04.24**	**Пн 27.10.25**	
1		**Дослідження проєкту та ініціація**	**18 дней**	**Пн 29.04.24**	**Ср 22.05.24**	
2		Обсяг проєкту	2 дней	Пн 29.04.24	Вт 30.04.24	
3		Бюджет	1 день	Пн 29.04.24	Пн 29.04.24	
4		Створення проджект owerview(Документ який описує проєкт загалом)	1 день	Пн 29.04.24	Пн 29.04.24	
5		Ідентифікація стейкхолдерів(RACI	5 дней	Вт 30.04.24	Пн 06.05.24	4
6		Підготовка беклогу	3 дней	Вт 07.05.24	Чт 09.05.24	5
7		Оцінка беклогу	1 день	Пт 10.05.24	Пт 10.05.24	6
8		Підготовка складу команди	4 дней	Пн 13.05.24	Чт 16.05.24	7
9		Підготовка JIRA flow(Опис статусів задач та їх послідовностей)	2 дней	Пт 17.05.24	Пн 20.05.24	8
10		Підготовка Definition of Done(Документ в якому описані критерії того що задача виконана)	2 дней	Вт 21.05.24	Ср 22.05.24	9

4.2.1 Дослідження проєкту та ініціація

Після закінчення етапу ‘‘Дослідження та ініціація’’ буде перехід до етапу ‘‘Організація роботи’’ . Цей етап поділений на дві частини, а саме ‘‘Підготовка інфраструктури проєкту’’ та ‘‘Підготовка команди’’.

У частині ‘‘Підготовка інфраструктури’’ будуть описані 4 завдання.

Для початку визначимо що таке інфраструктура IT-проэкту.

Інфраструктура проєкту - це сукупність апаратних та програмних засобів, середовищ і ресурсів, необхідних для реалізації та підтримки проєкту в області інформаційних технологій.

До цієї частини в рамках дипломного проєктування будуть віднесені такі завдання:

1.Пошук приміщення – для продуктивної роботи команди перш за все потрібно знайти приміщення. На цей етап призначено час виконання 7 днів. Цей час включає у себе пошук самого приміщення та весь процес оренди.

2.Підготовка програмного забезпечення – виконання цього пункту має важливе значення. Придбання ліцензійного програмного забезпечення буде гарантувати, що проєкт створення 3D графіки для комп’ютерної гри з тематикою виживання в скрутних умовах дотримується авторських прав та законодавства. Дотримання

цього пункту буде забезпечувати уникання потенційних проблем пов’язаних з порушенням авторських прав та безпеки. Також слід відмітити, що ліцензійне програмне забезпечення має постійну підтримку від розробників, що включає в себе оновлення та забезпечує безперебійну роботу ПО. Ще одним дуже важливим аспектом для проєкту ліцензійне ПО – отримання можливості удосконалення та інтегрування своїх розробок.

3.Підготовка обладнання - підготовка обладнання має велике значення і на це є декілька причин:

Забезпечення продуктивності команди: Правильно підібране нами та налаштоване обладнання буде дозволяти команді працювати ефективно та продуктивно. Швидке обладнання забезпечить швидку роботу з програмами розробки та іншими не менш важливими інструментами.

Забезпечення сумісності: Обладнання повинно бути сумісним з іншими системами. Це забезпечить швидку роботу співробітників, які працюють у онлайн режимі.

4.JIRA Setup – у цьому процесі ми будемо налаштовувати конфігурацію системи управління проєктом JIRA для оптимального використання в рамках нашого проєкту.

Далі перейдемо до другої частини етапу організації роботи (зображення 4.2.2) – ‘‘Підготовка команди’’. У цьому пункті будуть описані важливі етапи для ефективної роботи співробітників.

1.Створення вакансій та проведення співбесід – правильне створення вакансій та проведення співбесід допоможе нам скласти команду з відповідними навичками та досвідом, які потрібні для реалізації нашого проєкту.

2.Підготовка issue log - це документ який ми будемо використовувати в управлінні проєктом для реєстрації та відстеження проблем, помилок, ризиків, а також запитів на зміни, які будуть виникати протягом життєвого циклу нашого проєкту. Він буде включати в себе деталі про кожну проблему, такі як опис проблеми, дата її виникнення, пріоритет, статус, відповідальну особу, плановані дії та вирішення. Issue log буде допомагати нашій проєктній команді ефективно відстежувати та керувати всіма проблемами та ризиками.

3.Підготовка change request log - це інструмент управління, який ми будемо використовувати для реєстрації та відстеження запитів на зміни в нашому проэкті. Він буде містити деталі про кожен запит на зміну, такі як опис зміни, дата подання, пріоритет, статус, відповідальна особа, вплив на проєкт і підтвердження від замовника проєкту.

11		Організація роботи	13 дней	Чт 23.05.24	Пн 10.06.24	
12		Підготовка інфраструктур	11 дней	Чт 23.05.24	Чт 06.06.24	
13		Пошук приміщення	7 дней	Чт 23.05.24	Пт 31.05.24	1
14		Підготовка програмного забезпечення(закупівл	1 день	Пн 03.06.24	Пн 03.06.24	13
15		Підготовка обладнання	2 дней	Вт 04.06.24	Ср 05.06.24	14
16		JIRA setup	1 день	Чт 06.06.24	Чт 06.06.24	15
17		Підготовка команди	1 день	Вт 04.06.24	Вт 04.06.24	
18		Створення вакансій	1 день	Вт 04.06.24	Вт 04.06.24	14
19		Підготовка issue log (Документ в якому ведеться запис проблем проятгом проэкту)	2 дней	Пт 07.06.24	Пн 10.06.24	12
20		Підготовка change request log(Документ в якому ведеться список незапланованих запитів замовника протягом роботи над проєктом)	2 дней	Пт 07.06.24	Пн 10.06.24	12

4.2.2 Організація роботи

Отже, тепер є можливість переходити до самого важливого та відповідального етапу проєкту створення графіки для гри з тематикою виживання в скрутних умовах, а саме до етапу – ‘‘Розробка’’ (зображення 4.2.3).

До цього пункту були включені основні активності, пов’язані з розробкою програмного забезпечення та створення кінцевого продукту.

Основні етапи в межах проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах:

1.Створення програмного налаштування – це одна з основних задач та мета дипломного проєктування. В цій задачі буде проведена розробка власного забезпечення щоб в кінцевому результаті отримати код який буде працювати задля ефективнішого та продуктивного етапу розробки 3D оточення.

2.Інтеграція програмного налаштування в ПЗ – ця задача відповідає за інтеграцію власної розробки в програмне забезпечення, а саме в 3D редактор.

3.Тестування програмного налаштування – в цьому пункті буде проведено тестування нашої розробки задля безперебійної роботи.

4.Створення концепт-артів – в цьому процесі будуть розроблені візуальні концепції та макети, які будуть визначати зовнішній вигляд та атмосферу гри.

5.Створення ілюстрацій – ця задача несе відповідальність за створення ілюстрацій для локацій, предметів, подій тощо.

6.Моделювання об’єктів/персонажів/оточення – в цій задачі буде відбуватися процес створення віртуальних тривимірних об’єктів за допомогою 3D редактору та нашого розробленого програмного налаштування.

7.Створення матеріалів та текстур – в цьому процесі буде відбуватися створення візуальних елементів які будуть надавати об’єктам їх зовнішній вигляд у грі.

8.Створення анімації – ця задача відповідає за створення руху та динаміки об’єктів.

9.Тестування – завершальний етап у розробці кінцевого продукту, тестування відповідає за перевірку сумісності 3D елементів.

	Розробка	**360 днів**	**Вт 11.06.24**	**Пн 27.10.25**		
	Створення програмного налаштування	1 день	Пн 29.04.24	Пт 11.10.24		Розробник програмного забезпечення(Програміст
	Інтеграція програмного налаштування в ПЗ	1 день	Пн 14.10.24	Пт 22.11.24	22	Розробник програмного забезпечення(Програміст
	Тестування програмного налаштування	1 день	Пн 25.11.24	Вт 03.12.24	23	Розробник програмного забезпечення(Програміст
	Створення концепт-артів	30 днів	Ср 04.12.24	Вт 14.01.25	24	Concept Artist
	Створення ілюстрацій	1 день	Ср 15.01.25	Пн 03.02.25	25	2D Artist
	Моделювання об'єктів	1 день	Вт 04.02.25	Пн 17.03.25	26	Junior 3D Artist;Middle 3D
	Створення персонажів	30 днів	Вт 18.03.25	Пн 28.04.25	27	Character Artist
	Створення оточеня	1 день	Вт 29.04.25	Пн 09.06.25	28	Environment 3D Artist
	Створення матеріалів	60 днів	Вт 10.06.25	Пн 01.09.25	29	Material Artist
	Створення текстур	1 день	Вт 02.09.25	Пн 13.10.25	30	Middle 3D Artist;Senior 3D
	Створення анімації	60 днів	Вт 14.10.25	Пн 05.01.26	31	3D Animator
	Тестування	1 день	Вт 06.01.26	Пн 16.02.26	32	Тестувальник

4.2.3 Розробка

Важливо відмітити, усі описані завдання вище будуть виконуватись послідовно. Для кожного етапу визначені свої терміни та відповідальні співробітники проєкту.

4.3 Огляд ресурсів проєкту

У даній частині розділу будуть описані ресурси і бюджет проєкту. Тут будуть описані значення та важливості правильного управління ресурсами у проєкті створення створення 3D графіки для комп’ютерної гри з тематикою виживання в скрутних умовах. Можна зазначити, що ефективне використання ресурсів, таких як людські, матеріальні та фінансові, є ключовим аспектом успішного виконання будь-якого проєкту, зокрема в IT індустрії. Також можна нагадати, що огляд ресурсів проєкту допомагає зрозуміти, як ресурси будуть використовуватися

протягом проєкту, і як ефективно керувати для досягнення поставлених цілей та завдань.

Загальний бюджет проєкту буде складати 7.000.000 гривень.

На зображеннях 4.3.1 та 4.3.2 будуть наведені аркуші ресурсів та описані відповідальності кожного співробітнику проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах.

Слід зазначити, заробітня плата співробітників проєкту визначена за погодинну роботу, а робочий день триває 8 годин.

	Название ресурса	Тип	Одиниця вимірюван матеріалів	Ініціали	Група	Макс. единиц	Стандартна ставка	Ставка сверхурочн
1	Електроенергія	Витрати		E				
2	Оренда офісу	Витрати		O				
3	Обслуговування обладнання	Витрати		O				
4	Розробник програмного	Робота		P		100%	930,00 ₴/г	1 034,00 ₴/г
5	Concept Artist	Робота		C		100%	676,00 ₴/г	712,00 ₴/г
6	Junior 3D Artist	Робота		J		100%	121,30 ₴/г	135,00 ₴/г
7	Middle 3D Artist	Робота		M		100%	486,51 ₴/г	501,00 ₴/г
8	Senior 3D Artist	Робота		S		100%	786,00 ₴/г	804,00 ₴/г
9	Environment 3D Artist	Робота		E		100%	340,00 ₴/г	370,00 ₴/г
10	3D Generalist	Робота		3		100%	400,00 ₴/г	454,00 ₴/г
11	Material Artist	Робота		M		100%	332,00 ₴/г	397,00 ₴/г
12	Character Artist	Робота		C		100%	550,00 ₴/г	610,00 ₴/г
13	2D Artist	Робота		2		100%	410,00 ₴/г	489,00 ₴/г
14	3D Animator	Робота		3		100%	509,00 ₴/г	601,00 ₴/г

4.3.1 Аркуш ресурсів

1. Розробник програмного забезпечення(Програміст) – відповідає за розробку програмного налаштування в межах нашого проєкту. Реалізовує функціональність коду.

2.Concept Artist(Концептувальник) – відповідає за розробку візуальних концептів та ідей.

3.Junior/Middle/Senior 3D Artist - відповідають за створення тривимірних моделей, а саме: стилізовані об’єкти, об’єкти архітектури та інші асети. Їхні відповідальності можуть варіюватися в залежності від їх рівня досвіду і ролі в команді.

4.Environment 3D Artist(Художник оточення) – відповідає за створення оточення і локацій для проєкту, а саме: створення ландшафту, розміщення об’єктів, створення атмосферності.

5.3D Generalist – відповідає за широкий спектр завдань. Коротко цього співробітника можна описати ‘‘Людина швейцарьский ніж’’.

6.Material Artist – відповідальний за створення та оптимізацію матеріалів і текстур.

7.Character Artist – відповідає за створення тривимірних персонажів. У його обов'язки входить низка задач: концептуальний дизайн, моделювання, риггінг, анімація.

8.2D Artist – відповідає за створення двовимірних графічних елементів та розробку інтерфейсів.

9.3D Animator – відповідає за створення анімації тривимірних об'єктів та роботу зі скелетною анімацією.

15		HR Manager	Робота		Н		100%	127,00 ₴/г	214,00 ₴/г
16		QA Manager	Робота		Q		100%	136,00 ₴/г	241,00 ₴/г
17		Team Lead	Робота		Т		100%	1 127,00 ₴/г	1 300,00 ₴/г
18		Тестувальник	Робота		Т		100%	112,70 ₴/г	198,00 ₴/г
19		Керівник проєкту	Робота		К		100%	227,00 ₴/г	299,00 ₴/г
20		Проєктний менеджер	Робота		П		100%	563,80 ₴/г	647,00 ₴/г
21		Фінансовий менеджер	Робота		Ф		100%	164,40 ₴/г	210,00 ₴/г
22		Менеджер з управління якості	Робота		М		100%	211,30 ₴/г	297,00 ₴/г

4.3.2 Аркуш ресурсів

10.HR Manager – відповідальний за керування людськими ресурсами в проєкті, а саме: набір персоналу, онбординг, кадрове адміністрування.

11.QA Manager – менеджер з котролю якості та забезпечення якості програмного налаштування. В обов'язки менеджера з контролю якості входять: планування тестування, керування тестовим процесом, автоматизація тестування, встановлення стандартів якості, керування дефектами.

12.Team Lead – відповідальний за керівництво командою розробників, а саме: технічне керівництво, планування і контроль, взаємодія зі стейкхолдерами.

13.Тестувальник – відповідальний за якість програмного налаштування у межах нашого проєкту.

14.Керівник проєкту – людина яка відповідає за керування усіма етапами проєкту.

15.проєктний менеджер – відповідає за керування нашим проєктом з початку та до завершення. Має дуже різний пул обов'язків.

16.Фінансовий менеджер – має обов'язки ефективного керування фінансами та фінансовими питаннями проєкту з метою забезпечення його успішності.

17.Менеджер з управління якістю – відповідає за якість кінцевого продукту проєкту.

4.4 Аналіз зацікавлених сторін

Аналіз зацікавлених сторін (стейкхолдерів) для проєкту є обов’язковим етапом управління, тому в цьому підрозділі буде визначено, хто впливає на проєкт та на кого впливає проєкт. Це допоможе краще зрозуміти вимоги, очікування та потенційні ризики, пов'язані з кожним стейкхолдером. Для проєкту розробки 3D графіки для гри на тематику виживання в скрутних умовах буде виділено наступні групи зацікавлених сторін:

Внутрішні зацікавлені сторони проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах:

1. Команда розробників:

Інтереси: Виконати завдання відповідно до плану, отримати досвід та підвищення за успішність реалізації.

Очікування: Чітке розуміння вимог, наявність необхідних ресурсів, гарна комунікація з керівництвом.

Вплив: Впливають на якість та терміни виконання.

2.проєктний менеджер:

Інтереси: Успішне завершення проєкту в межах термінів та визначеного бюджету.

Очікування: Високий рівень комунікації з командою та керівництвом, чітке розуміння вимог.

Вплив: Впливає на ресурси, контроль виконання задач, координація команди.

3. Керівництво:

Інтереси: Підвищення конкурентоспроможності, збільшення прибутку в майбутньому.

Очікування: Відповідність проєкту зазначеним цілям.

Вплив: Визначає бюджет та ресурси, прийняття важливих рішень.

Зовнішні зацікавлені сторони проєкту створення 3D графіки для гри з тематикою виживання в скрутних умовах:

1.Постачальники ліцензійного програмного забезпечення:

Інтереси: Забезпечення використання програмних продуктів у проєкті.

Підтримка довгострокової співпраці.

Вплив: Впливають на доступність та умови використання програмного забезпечення.

2. Постачальники обладнання:

Інтереси: Збільшення продажу свого обладнання, підтримка довгострокової співпраці.

Вплив: Впливають на якість необхідного обладнання та матеріалів.

3. Інвестори:

Інтереси: Отримати максимальний прибуток від інвестованих коштів, забезпечення стабільності проєкту.

Вплив: Впливають на бюджет та ресурси проєкту.

4. Сертифікаційні органи:

Інтереси: Забезпечення відповідність продукту встановленим стандартам і нормам.

Вплив: Впливають на якість кінцевого продукту.

4.5 SWOT-аналіз

SWOT аналіз – це інструмент стратегічного планування, що використовується для оцінки внутрішніх та зовнішніх факторів, які впливають на організацію або проєкт. SWOT розшифровується як Strengths (Сильні сторони), Weaknesses (Слабкі сторони), Opportunities (Можливості) і Threats (Загрози).

Метою SWOT аналізу є ідентифікація ключових факторів, які можуть вплинути на успіх проєкту.

SWOT аналіз проєкту:

Сильні сторони	Слабкі сторони
Зростання популярності жанру: Ігри з тематикою виживання в скрутних умовах зарекомендували себе як один з найпопулярніших жанрів в індустрії. **Висока ступінь інновацій:** Ринок відомий своєю готовністю до впровадження нових технологій та ідей. **Широкі можливості монетизації:** Дозволяє розробникам використовувати різні моделі	**Зростання конкуренції:** З появою багатьох нових ігор у цьому жанрі конкуренція на ринку стає все більш жорсткою, що може ускладнити просування нових продуктів. **Технічні обмеження:** Можливі технічні обмеження при інтеграції різних інструментів та технологій. **Висока вартість розробки:**

монетизації, продаж базової гри, DLC, мікротранзакцій та підписок.	Значні витрати на ліцензійне програмне забезпечення та обладнання. **Тривалість процесу розробки:** Методологія розробки Waterfall може бути менш гнучкою, що впливає на адаптацію до змін під час розробки.
Можливості	**Загрози**
Розвиток нових технологій: Використання нових технологій для покращення якості графіки. **Партнерство:** Можливість співпраці з іншими компаніями та спеціалістами для покращення кінцевого продукту. **Розширення на нові ринки:** Можливість виходу на міжнародні ринки та залучення нового ком'юніті.	**Конкуренція з іншими жанрами:** Піратство та копіювання може призвести до зниження прибутку з ігрових копій. **Зміни в технологіях:** Швидкий розвиток технологій може зробити деякі інструменти і методи застарілими. **Фінансові ризики:** Нестача фінансування або непередбачені витрати можуть поставити проєкт під загрозу.

SWOT-аналіз проєкту створення 3D графіки для гри на тематику виживання в скрутних умовах показує, що проєкт має значний потенціал завдяки своїм сильним сторонам, таким як висока якість графіки та досвідчена команда. Проте, для успішної реалізації необхідно враховувати слабкі сторони, зокрема високі витрати та складність управління. Можливості зростання ринку ігор та розширення аудиторії відкривають перспективи для розвитку, проте необхідно бути готовими до конкуренції та технологічних ризиків.

Загалом, успіх проєкту залежатиме від здатності ефективно використовувати свої сильні сторони та можливості, водночас мінімізуючи вплив слабких сторін та загроз.

4.6 Політика якості проєкту створення 3D графіки для комп'ютерної гри з тематикою виживання в скрутних умовах.

проєкт зобов'язаний забезпечувати високу якість в розробці проєкту управління 3D графікою для гри шляхом впровадження та дотримання вимог стандарту ДСТУ ISO 9001:2015. проєкт прагне до постійного вдосконалення процесів та продуктів, щоб задовольнити потреби користувачів та забезпечити високий рівень якості у всіх аспектах діяльності.

Цілі політики якості проєкту створення 3D графіки для комп’ютерної гри з тематикою виживання в скрутних умовах:

1.Забезпечення відповідності продукту вимогам клієнтів та стандартам:

- Виконання всіх вимог, встановлених стандартам, а також вимог стандарту ДСТУ ISO 9001:2015.

- Постійне оновлення процесів та процедур для забезпечення відповідності вимогам.

2.Підвищення задоволеності користувачів:

- Забезпечення високої якості продукту, що відповідає очікуванням та потребам користувачів.

- Забезпечення вчасного виконання замовлень та вирішення будь-яких проблем, що виникають.

3.Постійне вдосконалення процесів:

- Впровадження системи управління якістю згідно з вимогами стандарту ISO 9001:2015.

- Постійне оцінювання та вдосконалення ефективності процесів розробки та управління проєктом.

4.Комунікація та співпраця:

- Забезпечення ефективної комунікації та співпраці між усіма зацікавленими сторонами проєкту.

- Вирішення конфліктів та проблем шляхом взаєморозуміння та співпраці.

5.Навчання та розвиток співробітників:

- Забезпечення навчання та розвитку персоналу для забезпечення високого рівня професійної компетентності.

- Створення сприятливої атмосфери для навчання та розвитку персоналу.

Ці цілі визначають нашу зобов'язаність досягти високого рівня якості у всіх аспектах розробки проєкту управління 3D графікою для гри відповідно до вимог стандарту ДСТУ ISO 9001:2015.

4.7 Висновки

1. Проведений загальний огляд проєкту.
2. Оглянуті та описані задачі проєкту.

3. Оглянуті та описані ресурси проєкту
4. Проаналізовані та виявлені зацікавлені сторони проєкту.
5. Проведений SWOT-аналіз проєкту.
6. Створено політику якості проєкту.

ВИСНОВОК

Управління проєктом розробки 3D графіки для комп’ютерної гри з тематикою виживання в скрутних умовах показало наскільки процес створення є складним та багатогранним, який вимагає глибоких знань у сфері програмування, 3D дизайну та управління проєктами. В рамках даного дипломного проектування було детально розглянуті всі ключові аспекти, від початкового планування проєкту та прийняття власних рішень до розробки концептуальних ідей та реалізації 3D моделей.

Розробка плагіну для Autodesk Maya на основі мови програмування Python дозволила значно підвищити продуктивність процесу моделювання, скоротити час на виконання рутинних завдань та забезпечити високу якість кінцевого продукту. Була обрана методологія управління проєктом Waterfall, вона забезпечила чітку структуру та послідовність виконання усіх завдань, що дозволило досягти поставлених цілей у визначені терміни.

Цей проєкт продемонстрував важливість використання сучасних інструментів та технологій розробки 3D графіки, а також підкреслив важливість значення управління проєктами для досягнення успішного результату.

Отримані знання та досвід можуть бути можуть бути використані для подальшого розвитку ігрової індустрії в Україні, сприяючи створенню інноваційних продуктів, які можуть стати важливими як на внутрішньому, так і на міжнародному ринку.

ДЖЕРЕЛА

ДОДАТКИ

Код плагіну у повному обсязі:

```
import maya.cmds as mc
import maya.mel as mm

if not mc.optionVar(ex='Game_Poly_Tools_averageLine_stepMode'):
    mc.optionVar(iv=['Game_Poly_Tools_averageLine_stepMode', 0])
if not mc.optionVar(ex='Game_Poly_Tools_flatten_axis'):
    mc.optionVar(sv=['Game_Poly_Tools_flatten_axis', 'view plane'])
    mc.symmetricModelling(e=True, t=0.01, st=0.01)
if not mc.optionVar(ex='Game_Poly_Tools_flatten_space'):
    mc.optionVar(sv=['Game_Poly_Tools_flatten_space', 'os'])
if not mc.optionVar(ex='Game_Poly_Tools_flatten_align'):
    mc.optionVar(sv=['Game_Poly_Tools_flatten_align', 'mean'])
if not mc.optionVar(ex='Game_Poly_Tools_relax_keepBorder'):
    mc.optionVar(iv=['Game_Poly_Tools_relax_keepBorder', 1])
if not mc.optionVar(ex='Game_Poly_Tools_relax_keepShape'):
    mc.optionVar(iv=['Game_Poly_Tools_relax_keepShape', 0])
if not mc.optionVar(ex='Game_Poly_Tools_relax_step'):
    mc.optionVar(iv=['Game_Poly_Tools_relax_step', 1])
if not mc.optionVar(ex='Game_Poly_Tools_randomize_space'):
    mc.optionVar(iv=['Game_Poly_Tools_randomize_space', 1])
if not mc.optionVar(ex='Game_Poly_Tools_randomize_axis_x'):
    mc.optionVar(iv=['Game_Poly_Tools_randomize_axis_x', 0])
if not mc.optionVar(ex='Game_Poly_Tools_randomize_axis_y'):
    mc.optionVar(iv=['Game_Poly_Tools_randomize_axis_y', 0])
if not mc.optionVar(ex='Game_Poly_Tools_randomize_axis_z'):
    mc.optionVar(iv=['Game_Poly_Tools_randomize_axis_z', 1])
if not mc.optionVar(ex='Game_Poly_Tools_randomize_shift'):
    mc.optionVar(fv=['Game_Poly_Tools_randomize_shift', 0.1])
if not mc.optionVar(ex='Game_Poly_Tools_randomize_keepBorder'):
    mc.optionVar(iv=['Game_Poly_Tools_randomize_keepBorder', 1])
if not mc.optionVar(ex='Game_Poly_Tools_averageLine_even'):
    mc.optionVar(iv=['Game_Poly_Tools_averageLine_even', 0])
if not mc.optionVar(ex='Game_Poly_Tools_averageLine_keepShape'):
    mc.optionVar(iv=['Game_Poly_Tools_averageLine_keepShape', 0])
if not mc.optionVar(ex='Game_Poly_Tools_averageLine_smooth'):
    mc.optionVar(iv=['Game_Poly_Tools_averageLine_smooth', 0])
if not mc.optionVar(ex='Game_Poly_Tools_averageLine_align'):
    mc.optionVar(iv=['Game_Poly_Tools_averageLine_align', 1])
if not mc.optionVar(ex='Game_Poly_Tools_averageLine_step'):
    mc.optionVar(iv=['Game_Poly_Tools_averageLine_step', 1])
if not mc.optionVar(ex='Game_Poly_Tools_circlelizeLine_even'):
    mc.optionVar(iv=['Game_Poly_Tools_circlelizeLine_even', 0])
if not mc.optionVar(ex='Game_Poly_Tools_circlelizeLine_align'):
    mc.optionVar(iv=['Game_Poly_Tools_circlelizeLine_align', 1])
if not mc.optionVar(ex='Game_Poly_Tools_circlelizeLine_radius'):
    mc.optionVar(iv=['Game_Poly_Tools_circlelizeLine_radius', 1])
if not mc.optionVar(ex='Game_Poly_Tools_circlelizeLine_customRadius'):
    mc.optionVar(fv=['Game_Poly_Tools_circlelizeLine_customRadius', 1])
if not mc.optionVar(ex='Game_Poly_Tools_straightenLine_even'):
    mc.optionVar(iv=['Game_Poly_Tools_straightenLine_even', 0])
if not mc.optionVar(ex='Game_Poly_Tools_shapeLine_even'):
    mc.optionVar(iv=['Game_Poly_Tools_shapeLine_even', 0])
if not mc.optionVar(ex='Game_Poly_Tools_mirror_mode'):
```

```
    mc.optionVar(iv=['Game_Poly_Tools_mirror_mode', 1])
if not mc.optionVar(ex='Game_Poly_Tools_duplicateEdges_degree'):
    mc.optionVar(iv=['Game_Poly_Tools_duplicateEdges_degree', 1])
if not mc.optionVar(ex='Game_Poly_Tools_connect_div'):
    mc.optionVar(iv=['Game_Poly_Tools_connect_div', 1])
if not mc.optionVar(ex='Game_Poly_Tools_insertEdgeLoop_div'):
    mc.optionVar(iv=['Game_Poly_Tools_insertEdgeLoop_div', 1])
if not mc.optionVar(ex='Game_Poly_Tools_subdivEdges_div'):
    mc.optionVar(iv=['Game_Poly_Tools_subdivEdges_div', 1])
if not mc.optionVar(ex='Game_Poly_Tools_mergeFacet_cleanVtxs'):
    mc.optionVar(iv=['Game_Poly_Tools_mergeFacet_cleanVtxs', 0])
if not mc.optionVar(ex='Game_Poly_Tools_selToPattern_reverse'):
    mc.optionVar(iv=['Game_Poly_Tools_selToPattern_reverse', 0])
if not mc.optionVar(ex='Game_Poly_Tools_selNonQuardFaces_3side'):
    mc.optionVar(iv=['Game_Poly_Tools_selNonQuardFaces_3side', 1])
if not mc.optionVar(ex='Game_Poly_Tools_selNonQuardFaces_5side'):
    mc.optionVar(iv=['Game_Poly_Tools_selNonQuardFaces_5side', 1])
if not mc.optionVar(ex='Game_Poly_Tools_selNonQuardFaces_6side'):
    mc.optionVar(iv=['Game_Poly_Tools_selNonQuardFaces_6side', 1])
if not mc.optionVar(ex='Game_Poly_Tools_selNonQuardStars_2star'):
    mc.optionVar(iv=['Game_Poly_Tools_selNonQuardStars_2star', 0])
if not mc.optionVar(ex='Game_Poly_Tools_selNonQuardStars_3star'):
    mc.optionVar(iv=['Game_Poly_Tools_selNonQuardStars_3star', 0])
if not mc.optionVar(ex='Game_Poly_Tools_selNonQuardStars_5star'):
    mc.optionVar(iv=['Game_Poly_Tools_selNonQuardStars_5star', 0])
if not mc.optionVar(ex='Game_Poly_Tools_selNonQuardStars_6star'):
    mc.optionVar(iv=['Game_Poly_Tools_selNonQuardStars_6star', 1])
if not mc.optionVar(ex='Game_Poly_Tools_selCenterVtxsThreshold'):
    mc.optionVar(fv=['Game_Poly_Tools_selCenterVtxsThreshold', 0.1])

import re, copy, os, glob
from random import uniform, randint, choice

mm.eval('source "dagMenuProc"')
mc.scriptJob(e=['Undo', '%s.pf_68()' % __name__])
commandDict = {'DeleteMeshHistory': '%s.delMeshHistory()' % __name__,
 'EditComponents_EditFacet': '%s.editFacet()' % __name__,
 'EditComponents_SpinDialog': '%s.spin(dialog=True)' % __name__,
 'EditComponents_SpinForward': '%s.spin()' % __name__,
 'EditComponents_SpinBackward': '%s.spin(reverse=True)' % __name__,
 'EditComponents_Connect': '%s.connect()' % __name__,
 'EditComponents_ConnectOptionBox': '%s.connectUI()' % __name__,
 'EditComponents_InsertEdgeLoop': '%s.insertEdgeLoop()' % __name__,
 'EditComponents_InsertEdgeLoopOptionBox': '%s.insertEdgeLoopUI()' % __name__,
 'EditComponents_SubdivideEdges': '%s.subdivEdges()' % __name__,
 'EditComponents_SubdivideEdgesOptionBox': '%s.subdivEdgesUI()' % __name__,
 'EditComponents_MergeFaces': '%s.mergeFacet()' % __name__,
 'EditComponents_MergeFacesOptionBox': '%s.mergeFacetUI()' % __name__,
 'EditComponents_AddNewFaces': '%s.addNewFaces()' % __name__,
 'MirrorPoly_CreateMirror': '%s.createMirror()' % __name__,
 'MirrorPoly_CreateMirrorOptionBox': '%s.mirrorPolyUI()' % __name__,
 'MirrorPoly_DeleteMirror': '%s.delMirror()' % __name__,
 'MirrorPoly_TakeHalf': '%s.takeHalf()' % __name__,
 'MirrorPoly_UpdateMirror': '%s.updateMirror()' % __name__,
 'MirrorPoly_ToggleMirror': '%s.toggleMirror()' % __name__,
 'MirrorPoly_SelectCenterVtxs': '%s.getCenterVtxs()' % __name__,
 'MirrorPoly_SelectCenterVtxsOptionBox': '%s.getCenterVtxsUI()' % __name__,
```

```
 'MirrorPoly_MoveToCenter': '%s.getCenterVtxsUI()' % __name__,
 'SplitMesh_Split': '%s.splitMesh()' % __name__,
 'SplitMesh_Merge': '%s.mergeMesh()' % __name__,
 'SplitMesh_Isolate': '%s.isolateSelFaces()' % __name__,
 'SplitMesh_Unisolate': '%s.unIsolateMesh()' % __name__,
 'SplitMesh_Duplicate': '%s.dupSelFaces()' % __name__,
 'Regularize_Flatten': '%s.flatten()' % __name__,
 'Regularize_FlattenOptionBox': '%s.flattenUI()' % __name__,
 'Regularize_Relax': '%s.relax()' % __name__,
 'Regularize_RelaxOptionBox': '%s.relaxUI()' % __name__,
 'Regularize_Randomize': '%s.randomize()' % __name__,
 'Regularize_RandomizeOptionBox': '%s.randomizeUI()' % __name__,
 'Regularize_AverageLine': '%s.averageLine()' % __name__,
 'Regularize_AverageLineOptionBox': '%s.averageLineUI()' % __name__,
 'Regularize_StraightenLine': '%s.straightenLine()' % __name__,
 'Regularize_StraightenLineOptionBox': '%s.straightenLineUI()' % __name__,
 'Regularize_CirclelizeLine': '%s.circlelizeLine()' % __name__,
 'Regularize_CirclelizeLineOptionBox': '%s.circlelizeLineUI()' % __name__,
 'Regularize_DuplicateEdges': '%s.duplicateEdges()' % __name__,
 'Regularize_DuplicateEdgesOptionBox': '%s.duplicateEdgesUI()' % __name__,
 'Regularize_ShapeLine': '%s.shapeLineUI()' % __name__,
 'Regularize_ShrinkWrap': '%s.shrinkWrapUI()' % __name__,
 'Select_QuickSel': '%s.quickSel()' % __name__,
 'Select_SelectNonQuardFaces': '%s.selNonQuardFaces()' % __name__,
 'Select_SelectNonQuardFacesOptionBox': '%s.selNonQuardFacesUI()' % __name__,
 'Select_SelectNonQuardStars': '%s.selNonQuardStars()' % __name__,
 'Select_SelectNonQuardStarsOptionBox': '%s.selNonQuardStarsUI()' % __name__,
 'Select_SelectOpenEdges': '%s.selBorder()' % __name__,
 'Select_ToPattern': '%s.selToPattern()' % __name__,
 'Select_ToPatternOptionBox': '%s.selToPatternUI()' % __name__,
 'Select_ToBorder': "%s.selToBorder(mode='border')" % __name__,
 'Select_ToInteral': "%s.selToBorder(mode='inside')" % __name__,
 'Display_ToggleShader': '%s.toggleShader()' % __name__,
 'Display_ToggleTransparency': '%s.toggleShaderTransparency()' % __name__,
 'Display_DisplayHardEdgesOnly': 'mc.polyOptions(ao=True,he=True)',
 'Display_DisplayAllEdges': 'mc.polyOptions(ao=True,ae=True)',
 'Display_HardenEdges': '%s.hardenEdge()' % __name__,
 'Display_SoftenEdges': '%s.softenEdge()' % __name__,
 'Game_Poly_Tools_info': '%s.info()' % __name__,
 'Game_Poly_Tools_register': '%s.register()' % __name__}

def setCmd():
    for namedCommand, command in commandDict.items():
        try:
            mc.runTimeCommand(namedCommand, c=command, cat='Game_Poly_Tools', cl='python')
        except:
            pass
def delCmd():
    for namedCommand in commandDict:
        try:
            mc.runTimeCommand(namedCommand, edit=True, delete=True)
        except:
            pass
def prtCmd():
    print '\n\n\n\n\n' + ver + ' nameCommands:\n'
    for i in sorted(commandDict.keys()):
        print i
```

```
setCmd()
mc.optionVar(iv=['Game_Poly_Tools_setNamedCommand', 1])
mc.optionVar(iv=['Game_Poly_Tools_setNamedCommand2', 1])

def create_menu():
    menus = mc.lsUI(menus=True)
    for eachMenu in menus:
        if 'Game_Poly_Tools' in mc.menu(eachMenu, q=True, l=True):
            mc.deleteUI(eachMenu)
            break

    melMenuCmds1 = '\n    menu -l "Game_Poly_Tools" -tearOff 1 -allowOptionBoxes 1 -p "MayaWindow";\n
menuItem -l "Delete Mesh History" -c "DeleteMeshHistory";\n    menuItem -d 1;\n\n    menuItem -l "Edit
Components" -subMenu 1 -tearOff 1 -allowOptionBoxes 1;\n    menuItem -l "Edit Facet" -c
"EditComponents_EditFacet";\n    menuItem -l "Spin" -subMenu 1 -tearOff 1 -allowOptionBoxes 1;\n
menuItem -l "Spin Dialog" -c "EditComponents_SpinDialog";\n    menuItem -l "-->" -c
"EditComponents_SpinForward";\n    menuItem -l "<--" -c "EditComponents_SpinBackward";\n    setParent -
menu ..;\n\n    menuItem -d 1;\n    menuItem -l "Connect" -c "EditComponents_Connect";\n    menuItem -
optionBox 1 -c "EditComponents_ConnectOptionBox";\n    menuItem -l "Insert Edge Loop" -c
"EditComponents_InsertEdgeLoop";\n    menuItem -optionBox 1 -c
"EditComponents_InsertEdgeLoopOptionBox";\n    menuItem -l "Subdivide Edges" -c
"EditComponents_SubdivideEdges";\n    menuItem -optionBox 1 -c
"EditComponents_SubdivideEdgesOptionBox";\n    menuItem -l "Merge Faces" -c
"EditComponents_MergeFaces";\n    menuItem -optionBox 1 -c "EditComponents_MergeFacesOptionBox";\n
menuItem -l "Add New Faces" -c "EditComponents_AddNewFaces";\n\n    setParent -menu ..;\n    menuItem -l
"MirrorPoly" -subMenu 1 -tearOff 1 -allowOptionBoxes 1;\n    menuItem -l "Create Mirror" -c
"MirrorPoly_CreateMirror";\n    menuItem -optionBox 1 -c "MirrorPoly_CreateMirrorOptionBox";\n
menuItem -l "Delete Mirror" -c "MirrorPoly_DeleteMirror";\n    menuItem -l "Take Half" -c
"MirrorPoly_TakeHalf";\n    menuItem -d 1;\n    menuItem -l "Update Mirror" -c "MirrorPoly_UpdateMirror";\n
menuItem -l "Toggle Mirror" -c "MirrorPoly_ToggleMirror";\n    menuItem -d 1;\n    menuItem -l "Sel Center
Vtxs" -c "MirrorPoly_SelectCenterVtxs";\n    menuItem -optionBox 1 -c
"MirrorPoly_SelectCenterVtxsOptionBox";\n    menuItem -l "Move to Center" -c
"MirrorPoly_MoveToCenter";\n\n    setParent -menu ..;\n    setParent -menu ..;\n    menuItem -l "Split Mesh" -
subMenu 1 -tearOff 1 -allowOptionBoxes 1;\n    menuItem -l "Split" -c "SplitMesh_Split";\n    menuItem -l
"Merge" -c "SplitMesh_Merge";\n    menuItem -d 1;\n    menuItem -l "Isolate" -c "SplitMesh_Isolate";\n
menuItem -l "Unisolate" -c "SplitMesh_Unisolate";\n    menuItem -d 1;\n    menuItem -l "Duplicate" -c
"SplitMesh_Duplicate";\n\n    setParent -menu ..;\n    menuItem -l "Regularize" -subMenu 1 -tearOff 1 -
allowOptionBoxes 1;\n\n    menuItem -l "Flatten" -c "Regularize_Flatten";\n    menuItem -optionBox 1 -c
"Regularize_FlattenOptionBox";\n\n    menuItem -l "Relax" -c "Regularize_Relax";\n    menuItem -optionBox 1
-c "Regularize_RelaxOptionBox";\n\n    menuItem -l "Randomize" -c "Regularize_Randomize";\n    menuItem -
optionBox 1 -c "Regularize_RandomizeOptionBox";\n\n    menuItem -l "Average Line" -c
"Regularize_AverageLine";\n    menuItem -optionBox 1 -c "Regularize_AverageLineOptionBox";\n\n
menuItem -l "Straighten Line" -c "Regularize_StraightenLine";\n    menuItem -optionBox 1 -c
"Regularize_StraightenLineOptionBox";\n\n    menuItem -l "Circlelize Line" -c "Regularize_CirclelizeLine";\n
menuItem -optionBox 1 -c "Regularize_CirclelizeLineOptionBox";\n\n    menuItem -l "Duplicate Edges" -c
"Regularize_DuplicateEdges";\n    menuItem -optionBox 1 -c "Regularize_DuplicateEdgesOptionBox";\n\n
menuItem -l "Shape Line..." -c "Regularize_ShapeLine";\n\n    menuItem -l "ShrinkWrap..." -c
"Regularize_ShrinkWrap";\n\n    setParent -menu ..;\n    menuItem -l "Select" -subMenu 1 -tearOff 1 -
allowOptionBoxes 1;\n    menuItem -l "Quick Sel" -c "Select_QuickSel";\n    menuItem -d 1;\n    menuItem -l
"Select NonQuard Faces" -c "Select_SelectNonQuardFaces";\n    menuItem -optionBox 1 -c
"Select_SelectNonQuardFacesOptionBox";\n    menuItem -l "Select NonQuard Stars" -c
"Select_SelectNonQuardStars";\n    menuItem -optionBox 1 -c "Select_SelectNonQuardStarsOptionBox";\n
menuItem -l "Select Open Edges" -c "Select_SelectOpenEdges";\n    menuItem -d 1;\n    menuItem -l "To
Pattern" -c "Select_ToPattern";\n    menuItem -optionBox 1 -c "Select_ToPatternOptionBox";\n    menuItem -l
"To Border" -c "Select_ToBorder";\n    menuItem -l "To Interal" -c "Select_ToInteral";\n\n    setParent -menu
..;\n    menuItem -l "Display" -subMenu 1 -tearOff 1 -allowOptionBoxes 1;\n    menuItem -l "Toggle Shader" -c
"Display_ToggleShader";\n    menuItem -l "Toggle Transparency" -c "Display_ToggleTransparency";\n
```

```
menuItem -d 1;\n    menuItem -l "Display Hard Edges Only" -c "Display_DisplayHardEdgesOnly";\n
menuItem -l "Display All Edges" -c "Display_DisplayAllEdges";\n    menuItem -d 1;\n    menuItem -l "Harden
Edges" -c "Display_HardenEdges";\n    menuItem -l "Soften Edges" -c "Display_SoftenEdges";\n\n    setParent
-menu ..;\n    menuItem -d 1;\n    menuItem -l "v1.62" -en 0;\n    '
   melMenuCmds = melMenuCmds1
   mm.eval(melMenuCmds)
create_menu()



def flattenUI():
   global flatten_RB_1
   global RBG_axis_flatten
   global flatten_RB_3
   global flatten_RB_4
   global flatten_RB_5
   global RCL_axis_flatten
   global flatten_RB_2
   global RBG_flatten_space
   global RBG_flatten_align
   pf_61()
   if pf_73('Game_Poly_Tools_flattenUI'):
      pf_71('Flatten Window Exists!')
   else:
      mc.window('Game_Poly_Tools_flattenUI', t='Flatten - Game_Poly_Tools', s=False, mxb=False)
      mc.columnLayout(co=['both', 5], rs=3, adj=True)
      RCL_axis_flatten = mc.rowColumnLayout(nc=6, cw=zip(range(1, 7), [50,
       35,
       35,
       35,
       80,
       100]), cal=zip(range(1, 7), ['left'] * 6))
      RBG_axis_flatten = mc.radioCollection()
      mc.text(l=' Axis:', al='left')
      flatten_RB_1 = mc.radioButton(l='X', sl=mc.optionVar(q='Game_Poly_Tools_flatten_axis') == 'x',
onc="mc.radioButtonGrp(%s.RBG_flatten_space,e=True,en=True);mc.optionVar(sv=['Game_Poly_Tools_flatt
en_axis','x'])" % __name__)
      flatten_RB_2 = mc.radioButton(l='Y', sl=mc.optionVar(q='Game_Poly_Tools_flatten_axis') == 'y',
onc="mc.radioButtonGrp(%s.RBG_flatten_space,e=True,en=True);mc.optionVar(sv=['Game_Poly_Tools_flatt
en_axis','y'])" % __name__)
      flatten_RB_3 = mc.radioButton(l='Z', sl=mc.optionVar(q='Game_Poly_Tools_flatten_axis') == 'z',
onc="mc.radioButtonGrp(%s.RBG_flatten_space,e=True,en=True);mc.optionVar(sv=['Game_Poly_Tools_flatt
en_axis','z'])" % __name__)
      flatten_RB_4 = mc.radioButton(l='View Plane', sl=mc.optionVar(q='Game_Poly_Tools_flatten_axis') ==
'view plane',
onc="mc.radioButtonGrp(%s.RBG_flatten_space,e=True,en=False);mc.optionVar(sv=['Game_Poly_Tools_flatt
en_axis','view plane'])" % __name__)
      flatten_RB_5 = mc.radioButton(l='Average Normal', sl=mc.optionVar(q='Game_Poly_Tools_flatten_axis')
== 'average normal',
onc="mc.radioButtonGrp(%s.RBG_flatten_space,e=True,en=False);mc.optionVar(sv=['Game_Poly_Tools_flatt
en_axis','average normal'])" % __name__)
      mc.setParent('..')
      RBG_flatten_space = mc.radioButtonGrp(l=' Space:', nrb=2, la2=['World Space', 'Object Space'], cw3=[50,
110, 120], cl3=['left'] * 3, sl=1 if mc.optionVar(q='Game_Poly_Tools_flatten_space') == 'ws' else 2,
en=mc.optionVar(q='Game_Poly_Tools_flatten_axis') in ('x', 'y', 'z'),
on1="mc.optionVar(sv=['Game_Poly_Tools_flatten_space','ws'])",
on2="mc.optionVar(sv=['Game_Poly_Tools_flatten_space','os'])")
```

```
        RBG_flatten_align = mc.radioButtonGrp(l=' Align:', nrb=3, la3=['Mean', 'Max', 'Min'], cw4=[50,
         80,
         70,
         70], cl4=['left'] * 4, sl=1 if mc.optionVar(q='Game_Poly_Tools_flatten_align') == 'mean' else (2 if
mc.optionVar(q='Game_Poly_Tools_flatten_align') == 'min' else 3),
on1="mc.optionVar(sv=['Game_Poly_Tools_flatten_align','mean'])",
on2="mc.optionVar(sv=['Game_Poly_Tools_flatten_align','min'])",
on3="mc.optionVar(sv=['Game_Poly_Tools_flatten_align','max'])")
        mc.button(l='Flatten', c='%s.flatten()' % __name__)
        mc.showWindow('Game_Poly_Tools_flattenUI')


def relaxUI():
    global CB_relax_keepBorder
    global CB_relax_keepShape
    global ISG_relax_step
    pf_61()
    if pf_73('Game_Poly_Tools_relaxUI'):
        pf_71('Relax Window Exists!')
    else:
        mc.window('Game_Poly_Tools_relaxUI', t='Relax', s=False, mxb=False)
        mc.columnLayout(co=['both', 5], rs=3, adj=True)
        mc.rowColumnLayout(nc=2, cw=[(1, 110), (2, 100)])
        CB_relax_keepBorder = mc.checkBox(l=' Keep Border', al='left',
v=mc.optionVar(q='Game_Poly_Tools_relax_keepBorder'),
cc="mc.optionVar(iv=['Game_Poly_Tools_relax_keepBorder',mc.checkBox(%s.CB_relax_keepBorder,q=True,
v=True)])" % __name__)
        CB_relax_keepShape = mc.checkBox(l=' Keep Shape', al='left',
v=mc.optionVar(q='Game_Poly_Tools_relax_keepShape'),
cc="mc.optionVar(iv=['Game_Poly_Tools_relax_keepShape',mc.checkBox(%s.CB_relax_keepShape,q=True,v
=True)])" % __name__)
        mc.setParent('..')
        ISG_relax_step = mc.intSliderGrp(l=' Step:', f=True, min=1, max=10, cw=[(1, 40), (2, 40)], cal=[(1, 'left')],
rat=[1, 'top', 3], v=mc.optionVar(q='Game_Poly_Tools_relax_step'),
cc="mc.optionVar(iv=['Game_Poly_Tools_relax_step',mc.intSliderGrp(%s.ISG_relax_step,q=True,v=True)])"
% __name__)
        mc.button(l='Relax', c='%s.relax()' % __name__)
        mc.showWindow('Game_Poly_Tools_relaxUI')


def averageLineUI():
    global CB_averageLine_even
    global CB_averageLine_keepShape
    global CB_averageLine_smooth
    global ISG_averageLine_step
    global RBG_averageLine_align
    pf_61()
    if pf_73('Game_Poly_Tools_averageLineUI'):
        pf_71('Average Line Window Exists!')
    else:
        mc.window('Game_Poly_Tools_averageLineUI', t='Average Line', s=False, mxb=False)
        mc.columnLayout(co=['both', 5], rs=3, adj=True)
        mc.rowColumnLayout(nc=3, cw=[(1, 80), (2, 120), (3, 80)])
        CB_averageLine_even = mc.checkBox(l=' Even', al='left',
v=mc.optionVar(q='Game_Poly_Tools_averageLine_even'),
cc="mc.optionVar(iv=['Game_Poly_Tools_averageLine_even',mc.checkBox(%s.CB_averageLine_even,q=True
```

```
,v=True)]);mc.intSliderGrp(%s.ISG_averageLine_step,e=True,en=not
mc.optionVar(q='Game_Poly_Tools_averageLine_even'))" % (__name__, __name__))
        CB_averageLine_keepShape = mc.checkBox(l=' Keep Shape', al='left',
v=mc.optionVar(q='Game_Poly_Tools_averageLine_keepShape'),
cc="mc.optionVar(iv=['Game_Poly_Tools_averageLine_keepShape',mc.checkBox(%s.CB_averageLine_keepS
hape,q=True,v=True)])" % __name__)
        CB_averageLine_smooth = mc.checkBox(l=' smooth', al='left',
v=mc.optionVar(q='Game_Poly_Tools_averageLine_smooth'),
cc="mc.optionVar(iv=['Game_Poly_Tools_averageLine_smooth',mc.checkBox(%s.CB_averageLine_smooth,q
=True,v=True)])" % __name__)
        mc.setParent('..')
        RBG_averageLine_align = mc.radioButtonGrp(l=' Align:', nrb=4, la4=['YZ',
         'XZ',
         'XY',
         'Auto'], cw5=[55,
         40,
         40,
         40,
         40], cl5=['left'] * 5, sl=mc.optionVar(q='Game_Poly_Tools_averageLine_align'),
cc="mc.optionVar(iv=['Game_Poly_Tools_averageLine_align',mc.radioButtonGrp(%s.RBG_averageLine_align
,q=True,sl=True)])" % __name__)
        ISG_averageLine_step = mc.intSliderGrp(l=' Step:', f=True, min=1, max=10, cw=[(1, 55), (2, 40)],
cal=[(1, 'left')], rat=[1, 'top', 3], en=not mc.optionVar(q='Game_Poly_Tools_averageLine_even'),
v=mc.optionVar(q='Game_Poly_Tools_averageLine_step'),
cc="mc.optionVar(iv=['Game_Poly_Tools_averageLine_step',mc.intSliderGrp(%s.ISG_averageLine_step,q=Tr
ue,v=True)])" % __name__)
        mc.button(l='Average Line', c='%s.averageLine()' % __name__)
        mc.showWindow('Game_Poly_Tools_averageLineUI')


def straightenLineUI():
    global CB_averageLine_even
    pf_61()
    if pf_73('Game_Poly_Tools_straightenLineUI'):
        pf_71('straighten Line Window Exists!')
    else:
        mc.window('Game_Poly_Tools_straightenLineUI', t='Straighten Line', s=False, mxb=False)
        mc.columnLayout(co=['both', 5], rs=3, adj=True)
        CB_averageLine_even = mc.checkBox(l=' Even', al='left',
v=mc.optionVar(q='Game_Poly_Tools_straightenLine_even'),
cc="mc.optionVar(iv=['Game_Poly_Tools_straightenLine_even',mc.checkBox(%s.CB_averageLine_even,q=Tr
ue,v=True)])" % __name__)
        mc.button(l='Straighten Line', w=230, c='%s.straightenLine()' % __name__)
        mc.showWindow('Game_Poly_Tools_straightenLineUI')


def circlelizeLineUI():
    global RBG_circlelizeLine_align
    global RBG_circlelizeLine_radius
    global CB_circlelizeLine_even
    global FF_circlelizeLine_customRadius
    pf_61()
    if pf_73('Game_Poly_Tools_circlelizeLineUI'):
        pf_71('Circlelize Line Window Exists!')
    else:
        mc.window('Game_Poly_Tools_circlelizeLineUI', t='Circlelize Line', s=False, mxb=False)
        mc.columnLayout(co=['both', 5], rs=3, adj=True)
```

```
        CB_circlelizeLine_even = mc.checkBox(l=' Even', al='left',
v=mc.optionVar(q='Game_Poly_Tools_circlelizeLine_even'),
cc="mc.optionVar(iv=['Game_Poly_Tools_circlelizeLine_even',mc.checkBox(%s.CB_circlelizeLine_even,q=T
rue,v=True)])" % __name__)
        mc.setParent('..')
        mc.columnLayout(rs=3, adj=True)
        RBG_circlelizeLine_align = mc.radioButtonGrp(l=' Align:', nrb=4, la4=['YZ',
         'XZ',
         'XY',
         'Auto'], cw5=[55,
         55,
         55,
         55,
         55], cl5=['left'] * 5, sl=mc.optionVar(q='Game_Poly_Tools_circlelizeLine_align'),
cc="mc.optionVar(iv=['Game_Poly_Tools_circlelizeLine_align',mc.radioButtonGrp(%s.RBG_circlelizeLine_al
ign,q=True,sl=True)])" % __name__)
        mc.rowColumnLayout(nc=2, cw=[(1, 270), (2, 60)], rat=[1, 'bottom', 3])
        RBG_circlelizeLine_radius = mc.radioButtonGrp(l=' Radius:', nrb=4, la4=['Mean',
         'Max',
         'Min',
         'Custom:'], cw5=[55,
         55,
         50,
         45,
         60], cl5=['left'] * 5, sl=mc.optionVar(q='Game_Poly_Tools_circlelizeLine_radius'),
cc="mc.optionVar(iv=['Game_Poly_Tools_circlelizeLine_radius',mc.radioButtonGrp(%s.RBG_circlelizeLine_r
adius,q=True,sl=True)]);mc.floatField(%s.FF_circlelizeLine_customRadius,e=True,en=(mc.optionVar(q='Gam
e_Poly_Tools_circlelizeLine_radius')==4))" % (__name__, __name__))
        FF_circlelizeLine_customRadius = mc.floatField(min=0.001, max=100, pre=2,
en=mc.optionVar(q='Game_Poly_Tools_circlelizeLine_radius') == 4,
v=mc.optionVar(q='Game_Poly_Tools_circlelizeLine_customRadius'),
cc="mc.optionVar(fv=['Game_Poly_Tools_circlelizeLine_customRadius',mc.floatField(%s.FF_circlelizeLine_c
ustomRadius,q=True,v=True)])" % __name__)
        mc.setParent('..')
        mc.button(l='Circlelize Line', c='%s.circlelizeLine()' % __name__)
        mc.showWindow('Game_Poly_Tools_circlelizeLineUI')


def shapeLineUI():
    global TFBG_shapeLine
    global BT_shapeLine
    global CB_shapeLine_even
    pf_61()
    if pf_73('Game_Poly_Tools_shapeLineUI'):
        pf_71('AverageLine Window Exists!')
    else:
        mc.window('Game_Poly_Tools_shapeLineUI', t='Shape Line', s=False, mxb=False)
        mc.columnLayout(co=['both', 5], rs=3, adj=True)
        CB_shapeLine_even = mc.checkBox(l=' Even', al='left',
v=mc.optionVar(q='Game_Poly_Tools_shapeLine_even'),
cc="mc.optionVar(iv=['Game_Poly_Tools_shapeLine_even',mc.checkBox(%s.CB_shapeLine_even,q=True,v=
True)])" % __name__)
        TFBG_shapeLine = mc.textFieldButtonGrp(eb=True, bl=' Add Curve ', ed=False, cw=[1, 180],
bc='%s.addShapeCurve()' % __name__)
        BT_shapeLine = mc.button(l='Shape Line', en=False, c='%s.shapeLine()' % __name__)
        mc.showWindow('Game_Poly_Tools_shapeLineUI')
```

```
def randomizeUI():
    global FSG_shift_randomize
    global RBG_space_randomize
    global CBG_axis_randomize
    global CB_protectBorder_randomize
    pf_61()
    if pf_73('Game_Poly_Tools_randomizeUI'):
        pf_71('Randomize Window Exists!')
    else:
        mc.window('Game_Poly_Tools_randomizeUI', t='Randomize', s=False, mxb=False)
        mc.columnLayout(co=['both', 5], rs=3, adj=True)
        RBG_space_randomize = mc.radioButtonGrp(l=' Space:', nrb=3, la3=['Normal', 'Object', 'World'],
cw4=[50,
         60,
         60,
         60], cl4=['left'] * 4, sl=mc.optionVar(q='Game_Poly_Tools_randomize_space'),
cc="mc.optionVar(iv=['Game_Poly_Tools_randomize_space',mc.radioButtonGrp(%s.RBG_space_randomize,q
=True,sl=True)])" % __name__)
        CBG_axis_randomize = mc.checkBoxGrp(l=' Axis:', ncb=3, la3=['X', 'Y', 'Z'], cw4=[50,
         60,
         60,
         60], cl4=['left'] * 4, v1=mc.optionVar(q='Game_Poly_Tools_randomize_axis_x'),
v2=mc.optionVar(q='Game_Poly_Tools_randomize_axis_y'),
v3=mc.optionVar(q='Game_Poly_Tools_randomize_axis_z'),
cc1="mc.optionVar(iv=['Game_Poly_Tools_randomize_axis_x',mc.checkBoxGrp(%s.CBG_axis_randomize,q=
True,v1=True)])" % __name__,
cc2="mc.optionVar(iv=['Game_Poly_Tools_randomize_axis_y',mc.checkBoxGrp(%s.CBG_axis_randomize,q=
True,v2=True)])" % __name__,
cc3="mc.optionVar(iv=['Game_Poly_Tools_randomize_axis_z',mc.checkBoxGrp(%s.CBG_axis_randomize,q=
True,v3=True)])" % __name__)
        FSG_shift_randomize = mc.floatSliderGrp(l=' Shift:', f=True,
v=mc.optionVar(q='Game_Poly_Tools_randomize_shift'), step=0.001, min=0.001, max=1, cw=[(1, 50), (2,
38)], cal=[(1, 'left')],
cc="mc.optionVar(fv=['Game_Poly_Tools_randomize_shift',mc.floatSliderGrp(%s.FSG_shift_randomize,q=Tr
ue,v=True)])" % __name__)
        CB_protectBorder_randomize = mc.checkBox(l='Keep Border',
v=mc.optionVar(q='Game_Poly_Tools_randomize_keepBorder'), al='left',
cc="mc.optionVar(iv=['Game_Poly_Tools_randomize_keepBorder',mc.checkBox(%s.CB_protectBorder_rando
mize,q=True,v=True)])" % __name__)
        mc.button(l='Randomize', c="%s.randomize()' % __name__)
        mc.showWindow('Game_Poly_Tools_randomizeUI')


def shrinkWrapUI():
    global CB_border_shrinkWrap
    global RBG_mode_shrinkWrap
    global BT_shrinkWrap
    global TFBG_shrinkWrap
    pf_61()
    if pf_73('Game_Poly_Tools_shrinkWrapUI'):
        pf_71('ShrinkWrap Window Exists!')
    else:
        mc.window('Game_Poly_Tools_shrinkWrapUI', t='ShrinkWrap', s=False, mxb=False)
        mc.columnLayout(co=['both', 5], rs=3, adj=True)
        TFBG_shrinkWrap = mc.textFieldButtonGrp(eb=True, bl=' Add Target ', ed=False, cw=[1, 150],
bc='%s.addWrapTarget()' % __name__)
```

```
        mc.rowColumnLayout(nc=2, cw=[(1, 150), (2, 60)], cat=[2, 'left', 12])
        RBG_mode_shrinkWrap = mc.radioButtonGrp(nrb=2, l=' Mode:', la2=['Vertex', 'Facet'], sl=1, cw3=[45,
60, 45], cal=[1, 'left'], on1='mc.checkBox(%s.CB_border_shrinkWrap,e=True,en=False)' % __name__,
on2='mc.checkBox(%s.CB_border_shrinkWrap,e=True,en=True)' % __name__)
        CB_border_shrinkWrap = mc.checkBox(l='Border', v=True, en=False)
        mc.setParent('..')
        BT_shrinkWrap = mc.button(l='ShrinkWrap', en=False, c='%s.shrinkWrap()' % __name__)
        mc.showWindow('Game_Poly_Tools_shrinkWrapUI')


def shrinkWrapUI_popupMenu():
    if pf_73('shrinkWrapUI_popupMenu'):
        mc.deleteUI('shrinkWrapUI_popupMenu')
    target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
    if target and mc.objExists(target):
        mc.popupMenu('shrinkWrapUI_popupMenu', p=TFBG_shrinkWrap)
        mc.menuItem(l='Enable', cb=True, c='%s.enableShrinkWrap()' % __name__)
        mc.menuItem(d=True)
        mc.menuItem(l='Select Target', c='%s.selWrapTarget()' % __name__)
        mc.menuItem(l='Remove Target', c='%s.removeWrapTarget()' % __name__)
        mc.menuItem(d=True)
        mc.menuItem(l='Make Target Live', c="mc.makeLive('%s')" % mc.listRelatives(target, s=True)[0])
        mc.menuItem(l='Turn Off Live', c='mc.makeLive(n=True)')
        mc.menuItem(d=True)
        mc.menuItem(l='Target Visibility', cb=mc.getAttr(target + '.v'), c='%s.toggleWrapTargetVis()' %
__name__)
        targetShape = mc.listRelatives(target, s=True)[0]
        curShaderGrp = mc.listConnections(targetShape, t='shadingEngine')
        if curShaderGrp:
            curShader = mc.listConnections(curShaderGrp[0], t='lambert')
            if curShader:
                curShader = curShader[0]
                transparency = any(mc.getAttr(curShader + '.transparency')[0])
        mc.menuItem(l='Target Transparency', cb=transparency, c='%s.toggleWrapTargetTranspency()' %
__name__)
        mc.radioMenuItemCollection(p='shrinkWrapUI_popupMenu')
        mc.menuItem(l='Normal', rb=True if not mc.getAttr(target + '.overrideEnabled') or
mc.getAttr('%s.overrideDisplayType' % target) == 0 else False, c='%s.toggleWrapTargetOverride(mode=0)' %
__name__)
        mc.menuItem(l='Template', rb=True if mc.getAttr('%s.overrideDisplayType' % target) == 1 and
mc.getAttr(target + '.overrideEnabled') else False, c='%s.toggleWrapTargetOverride(mode=1)' % __name__)
        mc.menuItem(l='Reference', rb=True if mc.getAttr('%s.overrideDisplayType' % target) == 2 and
mc.getAttr(target + '.overrideEnabled') else False, c='%s.toggleWrapTargetOverride(mode=2)' % __name__)


def relax():
    pf_61()
    selComps = pf_06()
    if selComps:
        selComps, selVtxs = selComps[:2]
        keepBorder = mc.optionVar(q='Game_Poly_Tools_relax_keepBorder')
        keepShape = mc.optionVar(q='Game_Poly_Tools_relax_keepShape')
        step = mc.optionVar(q='Game_Poly_Tools_relax_step')
        if keepBorder:
            selVtxs = [ vtx for vtx in selVtxs if not pf_08(vtx) ]
        if keepShape:
            mesh = pf_29()
```

```
            target = None
            if pf_73('Game_Poly_Tools_shrinkWrapUI') and mc.button(BT_shrinkWrap, q=True, en=True):
                target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
                if target:
                    if mc.objExists(target):
                        targetMesh = target
                    else:
                        target = None
            if not target:
                targetMesh = mc.duplicate(mesh)[0]
                mc.setAttr(targetMesh + '.v', False)
        if selVtxs:
            gMainProgressBar = mm.eval('$tmp=$gMainProgressBar')
            mc.progressBar(gMainProgressBar, edit=True, beginProgress=True, isInterruptable=True,
status='relaxing...', maxValue=step * 5 + 5)
            for i in range(step):
                if mc.progressBar(gMainProgressBar, query=True, isCancelled=True):
                    break
                mc.progressBar(gMainProgressBar, edit=True, step=1)
                mc.polyAverageVertex(selVtxs)
                delMeshHistory()
                mc.progressBar(gMainProgressBar, edit=True, step=4)

            if keepShape:
                pf_48(targetMesh, mesh, len(selVtxs), [ pf_32(vtx) for vtx in selVtxs ])
            if keepShape and not target:
                mc.delete(targetMesh)
            mc.progressBar(gMainProgressBar, edit=True, step=2)
            mc.select(selComps)
            pf_68()
            mc.progressBar(gMainProgressBar, edit=True, step=3)
            delMeshHistory()
            pf_22()
            mc.progressBar(gMainProgressBar, edit=True, endProgress=True)


def flatten():
    pf_61()
    selFaces = mc.filterExpand(sm=34)
    selMode = pf_06('vtx')
    if selMode and selMode[1]:
        selComps, vtxs = selMode[0], selMode[1]
        vtxNum = len(vtxs)
        mesh = pf_29()
        axis = mc.optionVar(q='Game_Poly_Tools_flatten_axis')
        space = mc.optionVar(q='Game_Poly_Tools_flatten_space')
        align = mc.optionVar(q='Game_Poly_Tools_flatten_align')
        if axis in ('average normal', 'view plane'):
            if axis == 'average normal':
                if selFaces:
                    normal = pf_37(selFaces).toTuple()
                else:
                    pf_71('No faces selectd!')
                    return False
            elif axis == 'view plane':
                modelPanel = mc.getPanel(wf=1)
                camera = mc.modelPanel(modelPanel, q=True, camera=True)
```

```
            targetNull = mc.group(em=True)
            baseNull = mc.group(em=True)
            mc.move(0, 0, 1, targetNull)
            mc.parent(targetNull, baseNull)
            mc.delete(mc.orientConstraint(camera, baseNull))
            normal = mc.xform(targetNull, q=True, ws=True, rp=True)
            mc.delete(baseNull)
         centerPos = mc.xform(mesh, q=True, ws=True, rp=True)
         parentNull = mc.group(em=True)
         mc.xform(parentNull, ws=True, t=centerPos)
         mc.parent(mesh, parentNull)
         loc = mc.spaceLocator()
         mc.xform(loc, ws=True, t=centerPos)
         mc.xform(loc, ws=True, r=True, t=(0, 1, 0))
         mc.delete(mc.aimConstraint(loc, parentNull, aim=normal), loc)
         space = 'ws'
         axis2 = 'y'
      else:
         axis2 = axis
      funcDict = {'mean': 'sum',
       'max': 'max',
       'min': 'min'}
      vtxPosGroup = pf_69.vectorList(eval('mc.xform(vtxs,q=True,%s=True,t=True)' % space))
      num = vtxNum if align == 'mean' else 1
      val = eval('%s([pos.%s for pos in vtxPosGroup])/%s' % (funcDict[align], axis2, num))
      eval('mc.move(val,vtxs,%s=True,%s=True)' % (space, axis2))
      if axis in ('average normal', 'view plane'):
         mc.setAttr(parentNull + '.r', 0, 0, 0)
         mc.ungroup(parentNull)
      mm.eval('doMenuComponentSelection("%s", "facet")' % mesh)
      mc.select(selComps)
      pf_70()


def randomize():
   pf_61()
   selComps = pf_06('vtx')
   if selComps:
      comps, vtxs, selMode = selComps
      if mc.optionVar(q='Game_Poly_Tools_randomize_keepBorder'):
         vtxs = [ vtx for vtx in vtxs if not pf_08(vtx) ]
      if vtxs:
         gMainProgressBar = mm.eval('$tmp=$gMainProgressBar')
         mc.progressBar(gMainProgressBar, edit=True, beginProgress=True, isInterruptable=True,
status='Randomizing...', maxValue=len(vtxs) * 1.1)
         space = mc.optionVar(q='Game_Poly_Tools_randomize_space')
         shift = mc.optionVar(q='Game_Poly_Tools_randomize_shift')
         axis_x = mc.optionVar(q='Game_Poly_Tools_randomize_axis_x')
         axis_y = mc.optionVar(q='Game_Poly_Tools_randomize_axis_y')
         axis_z = mc.optionVar(q='Game_Poly_Tools_randomize_axis_z')
         cmd = 'mc.polyMoveVertex(vtx,'
         if space == 3:
            cmd += 'ws=True,'
         if axis_x:
            if space == 1:
               cmd += 'ltx=uniform(%f,%f),' % (-shift, shift)
            else:
```

```
                cmd += 'tx=uniform(%f,%f),' % (-shift, shift)
            if axis_y:
                if space == 1:
                    cmd += 'lty=uniform(%f,%f),' % (-shift, shift)
                else:
                    cmd += 'ty=uniform(%f,%f),' % (-shift, shift)
            if axis_z:
                if space == 1:
                    cmd += 'ltz=uniform(%f,%f),' % (-shift, shift)
                else:
                    cmd += 'tz=uniform(%f,%f),' % (-shift, shift)
            cmd = cmd[:-1]
            cmd += ')'
            for vtx in vtxs:
                if mc.progressBar(gMainProgressBar, query=True, isCancelled=True):
                    break
                eval(cmd)
                delMeshHistory()
                mc.progressBar(gMainProgressBar, edit=True, step=1)

            mc.select(comps)
            mc.progressBar(gMainProgressBar, edit=True, endProgress=True)
            pf_22()


def straightenLine():
    pf_61()
    selEdges = mc.filterExpand(sm=32)
    selVtxs = mc.filterExpand(sm=31)
    if selEdges and len(selEdges) > 1:
        vtxGrps = pf_03(selEdges, breakVtxs=selVtxs)
        if vtxGrps:
            openVtxGrps = vtxGrps[0]
            if openVtxGrps:
                grpsNum = len(openVtxGrps)
                mesh = pf_29()
                even = mc.optionVar(q='Game_Poly_Tools_straightenLine_even')
                gMainProgressBar = mm.eval('$tmp=$gMainProgressBar')
                mc.progressBar(gMainProgressBar, edit=True, beginProgress=True, isInterruptable=True,
status='straightening...', maxValue=grpsNum)
                for vtxGrp in openVtxGrps:
                    if mc.progressBar(gMainProgressBar, query=True, isCancelled=True):
                        break
                    pf_43(vtxGrp, even)
                    mc.progressBar(gMainProgressBar, edit=True, step=1)

                mm.eval('doMenuComponentSelection("%s", "edge")' % mesh)
                mc.select(selEdges)
                if selVtxs:
                    mc.select(selVtxs, add=True)
                mc.progressBar(gMainProgressBar, edit=True, endProgress=True)
                delMeshHistory()
                pf_22()


def circlelizeLine():
    pf_61()
```

```
    selEdges = mc.filterExpand(sm=32)
    if selEdges and len(selEdges) > 1:
        vtxGrps = pf_03(selEdges)
        if vtxGrps:
            closeVtxGrps = vtxGrps[1]
            if closeVtxGrps:
                even = mc.optionVar(q='Game_Poly_Tools_circlelizeLine_even')
                grpsNum = len(closeVtxGrps)
                mesh = pf_29()
                alignDict = {1: 'YZ',
                 2: 'XZ',
                 3: 'XY',
                 4: 'Auto'}
                align = alignDict[mc.optionVar(q='Game_Poly_Tools_circlelizeLine_align')]
                radiusDict = {1: 'Mean',
                 2: 'Max',
                 3: 'Min',
                 4: 'Custom'}
                radius = radiusDict[mc.optionVar(q='Game_Poly_Tools_circlelizeLine_radius')]
                radiusValue = mc.optionVar(q='Game_Poly_Tools_circlelizeLine_customRadius')
                gMainProgressBar = mm.eval('$tmp=$gMainProgressBar')
                mc.progressBar(gMainProgressBar, edit=True, beginProgress=True, isInterruptable=True,
status='circlelizing...', maxValue=grpsNum)
                for vtxGrp in closeVtxGrps:
                    if mc.progressBar(gMainProgressBar, query=True, isCancelled=True):
                        break
                    pf_42(vtxGrp, even, radius, radiusValue, align, mesh)
                    mc.progressBar(gMainProgressBar, edit=True, step=1)

                mm.eval('doMenuComponentSelection("%s", "edge")' % mesh)
                mc.select(selEdges)
                mc.progressBar(gMainProgressBar, edit=True, endProgress=True)
                delMeshHistory()
                pf_22()


def averageLine():
    pf_61()
    selEdges = mc.filterExpand(sm=32)
    selVtxs = mc.filterExpand(sm=31)
    if selEdges and len(selEdges) > 1:
        vtxGrps = pf_03(selEdges, breakVtxs=selVtxs)
        if vtxGrps:
            even = mc.optionVar(q='Game_Poly_Tools_averageLine_even')
            keepShape = mc.optionVar(q='Game_Poly_Tools_averageLine_keepShape')
            openVtxGrps, closeVtxGrps = vtxGrps
            allVtxGrps = openVtxGrps + closeVtxGrps
            grpsNum = len(allVtxGrps)
            mesh = pf_29()
            if keepShape:
                target = None
                if pf_73('Game_Poly_Tools_shrinkWrapUI') and mc.button(BT_shrinkWrap, q=True, en=True):
                    target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
                    if target:
                        if mc.objExists(target):
                            targetMesh = target
                        else:
```

```
                target = None
            if not target:
                targetMesh = mc.duplicate(mesh)[0]
                mc.setAttr(targetMesh + '.v', False)
        smooth = mc.optionVar(q='Game_Poly_Tools_averageLine_smooth')
        alignDict = {1: 'YZ',
         2: 'XZ',
         3: 'XY',
         4: 'Auto'}
        align = alignDict[mc.optionVar(q='Game_Poly_Tools_averageLine_align')]
        step = mc.optionVar(q='Game_Poly_Tools_averageLine_step')
        gMainProgressBar = mm.eval('$tmp=$gMainProgressBar')
        mc.progressBar(gMainProgressBar, edit=True, beginProgress=True, isInterruptable=True,
status='averaging...', maxValue=grpsNum)
        for vtxGrp in allVtxGrps:
            if mc.progressBar(gMainProgressBar, query=True, isCancelled=True):
                break
            pf_44(vtxGrp, vtxGrp in closeVtxGrps, mesh, even, smooth, step, align)
            if keepShape:
                pf_48(targetMesh, mesh, len(vtxGrp), [ pf_32(vtx) for vtx in vtxGrp ])
            mc.progressBar(gMainProgressBar, edit=True, step=1)

        if keepShape and not target:
            mc.delete(targetMesh)
        mm.eval('doMenuComponentSelection("%s", "edge")' % mesh)
        mc.select(selEdges)
        if selVtxs:
            mc.select(selVtxs, add=True)
        mc.progressBar(gMainProgressBar, edit=True, endProgress=True)
        delMeshHistory()
        pf_22()


def pf_45(vtxGrp, close, mesh, even):
    refreshShapeCurve()
    shapeCurve = mc.textFieldButtonGrp(TFBG_shapeLine, q=True, tx=True)
    if shapeCurve and mc.objExists(shapeCurve):
        vtxNum = len(vtxGrp)
        if even:
            dupCurve = mc.duplicate(shapeCurve)[0]
            dupCurve = mc.rebuildCurve(dupCurve, s=vtxNum if close else vtxNum - 1, d=3 if close else 1,
ch=False)[0]
            curveStartPos = pf_69(mc.xform('%s.ep[%s]' % (dupCurve, 0), q=True, ws=True, t=True))
            if close:
                firstVtxId = 0
                firstVtxPos = pf_69(mc.xform(vtxGrp[0], q=True, ws=True, t=True))
                firstVtxDist = curveStartPos.distance(firstVtxPos)
                for i in range(1, vtxNum):
                    vtxPos = pf_69(mc.xform(vtxGrp[i], q=True, ws=True, t=True))
                    dist = curveStartPos.distance(vtxPos)
                    if dist < firstVtxDist:
                        firstVtxDist = dist
                        firstVtxId = i

                vtxGrp = vtxGrp[firstVtxId:] + vtxGrp[:firstVtxId]
                curveEp2Pos = pf_69(mc.xform('%s.ep[%s]' % (dupCurve, 1), q=True, ws=True, t=True))
                vtx2Pos = pf_69(mc.xform(vtxGrp[1], q=True, ws=True, t=True))
```

```
            vtx_last_Pos = pf_69(mc.xform(vtxGrp[-1], q=True, ws=True, t=True))
            if curveEp2Pos.distance(vtx2Pos) > curveEp2Pos.distance(vtx_last_Pos):
                vtxGrp.reverse()
                vtxGrp = vtxGrp[-1:] + vtxGrp[:-1]
        else:
            firstVtxPos = pf_69(mc.xform(vtxGrp[0], q=True, ws=True, t=True))
            lastVtxPos = pf_69(mc.xform(vtxGrp[-1], q=True, ws=True, t=True))
            firstDist = curveStartPos.distance(firstVtxPos)
            lastDist = curveStartPos.distance(lastVtxPos)
            if firstDist > lastDist:
                vtxGrp.reverse()
        for i in range(vtxNum):
            mc.xform(vtxGrp[i], ws=True, t=mc.xform('%s.ep[%s]' % (dupCurve, i), q=True, ws=True, t=True))

        mc.delete(dupCurve)
        pf_48(shapeCurve, mesh, vtxNum, [ pf_32(vtx) for vtx in vtxGrp ])
    else:
        pf_48(shapeCurve, mesh, vtxNum, [ pf_32(vtx) for vtx in vtxGrp ])


def shapeLine():
    pf_61()
    if pf_73('Game_Poly_Tools_shapeLineUI'):
        selEdges = mc.filterExpand(sm=32)
        if selEdges:
            continua = pf_17(selEdges, returnMode='comps')
            if continua:
                close = True if continua[0] == 'close' else False
                vtxGrp = continua[1]
                mesh = pf_29()
                even = mc.optionVar(q='Game_Poly_Tools_shapeLine_even')
                pf_45(vtxGrp, close, mesh, even)
                mc.select(selEdges)
                mm.eval('doMenuComponentSelection("%s", "edge")' % mesh)


def shrinkWrap():
    pf_61()
    if pf_73('Game_Poly_Tools_shrinkWrapUI'):
        refreshWrapTarget()
        target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
        if target and mc.objExists(target):
            mesh = pf_29()
            if mesh and mesh != target:
                delMeshHistory()
                pf_22()
                shrinkMode = mc.radioButtonGrp(RBG_mode_shrinkWrap, q=True, sl=True)
                selComps = pf_06('selModeOnly')
                if selComps:
                    if shrinkMode == 1:
                        selComps, selVtxs, selMode = pf_06('vtx')
                        vtxNum = len(selVtxs)
                        allVtxIds = [ pf_32(vtx) for vtx in selVtxs ]
                    elif shrinkMode == 2:
                        selComps, selFaces, selMode = pf_06('face')
                        faceNum = len(selFaces)
                        allFaceIdsSet_1 = set([ pf_32(face) for face in selFaces ])
```

```
            else:
                msg = mc.confirmDialog(t='Warnning!', m='Will ShrinkWrap the whole Mesh, are you sure?',
b=['OK', 'Cancel'], db='OK', ds='Cancel', cb='Cancel')
                if msg == 'OK':
                    if shrinkMode == 1:
                        selComps, selMode = (None, 'mesh')
                        vtxNum = mc.polyEvaluate(mesh, v=True)
                        allVtxIds = range(vtxNum)
                    elif shrinkMode == 2:
                        selComps, selMode = (None, 'mesh')
                        faceNum = mc.polyEvaluate(mesh, f=True)
                        allFaceIdsSet_1 = set(range(faceNum))
                else:
                    return False
            targetFaceNum = mc.polyEvaluate(target, f=True)
            disableUndo = False
            if targetFaceNum > 5000 and (shrinkMode == 1 and vtxNum > 500 or shrinkMode == 2 and faceNum
> 500):
                disableUndo = True
                mc.undoInfo(swf=False)
            if shrinkMode == 1:
                pf_48(target, mesh, vtxNum, allVtxIds, progressBar=True)
            elif shrinkMode == 2:
                pf_46(target, mesh, faceNum, allFaceIdsSet_1, progressBar=True)
                fixBorder = mc.checkBox(CB_border_shrinkWrap, q=True, v=True)
                if fixBorder:
                    if selComps:
                        selEdges = pf_06('edge')[1]
                        selBorderEdges = [ edge for edge in selEdges if pf_07(edge) ]
                    else:
                        selBorderEdges = pf_23(mesh)
                    if selBorderEdges:
                        pf_47(target, mesh, selBorderEdges, progressBar=True)
            if selMode == 'mesh':
                mc.select(mesh)
            else:
                mm.eval('doMenuComponentSelection("%s", "%s")' % (mesh, selMode))
                mc.select(selComps)
            if disableUndo:
                mc.undoInfo(swf=True)


def enableShrinkWrap():
    state = mc.button(BT_shrinkWrap, q=True, en=True)
    mc.button(BT_shrinkWrap, e=True, en=not state)


def refreshWrapTarget():
    target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
    if not mc.objExists(target):
        removeWrapTarget()
        pf_71('%s not exists!' % target)


def selWrapTarget():
    target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
    if mc.objExists(target):
```

```
        mc.select(target)
    refreshWrapTarget()


def removeWrapTarget():
    mc.textFieldButtonGrp(TFBG_shrinkWrap, e=True, tx='')
    mc.button(BT_shrinkWrap, e=True, en=False)
    shrinkWrapUI_popupMenu()


def toggleWrapTargetVis():
    target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
    if mc.objExists(target):
        state = mc.getAttr(target + '.v')
        mc.setAttr(target + '.v', not state)
    refreshWrapTarget()


def toggleWrapTargetTranspency():
    target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
    if mc.objExists(target):
        targetShape = mc.listRelatives(target, s=True)[0]
        curShaderGrp = mc.listConnections(targetShape, t='shadingEngine')
        if curShaderGrp:
            curShader = mc.listConnections(curShaderGrp[0], t='lambert')
            if curShader:
                curShader = curShader[0]
                transparency = any(mc.getAttr(curShader + '.transparency')[0])
                if transparency:
                    mc.setAttr(curShader + '.transparency', 0, 0, 0)
                else:
                    mc.setAttr(curShader + '.transparency', 0.3, 0.3, 0.3)
    refreshWrapTarget()


def toggleWrapTargetOverride(mode = 0):
    target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
    if mc.objExists(target):
        state = mc.getAttr(target + '.overrideEnabled')
        if not state:
            mc.setAttr(target + '.overrideEnabled', not state)
        mc.setAttr(target + '.overrideDisplayType', mode)
    refreshWrapTarget()


def addWrapTarget():
    sel = mc.ls(sl=True)
    target = None
    if sel:
        if mc.nodeType(sel[0]) == 'transform':
            shape = mc.listRelatives(sel[0], s=True)[0]
            if mc.objectType(shape, isa='geometryShape'):
                target = sel[0]
            else:
                target = None
        elif mc.objectType(sel[0], isa='geometryShape') and '.' not in sel[0]:
            transform = mc.listRelatives(sel[0], p=True)[0]
```

```
            target = transform
    if target:
        mc.textFieldButtonGrp(TFBG_shrinkWrap, e=True, tx=target)
        mc.button(BT_shrinkWrap, e=True, en=True)
    else:
        mc.textFieldButtonGrp(TFBG_shrinkWrap, e=True, tx='')
        mc.button(BT_shrinkWrap, e=True, en=False)
    mc.refresh()
    shrinkWrapUI_popupMenu()


def addShapeCurve():
    sel = mc.ls(sl=True)
    target = None
    if sel:
        if mc.nodeType(sel[0]) == 'transform':
            shape = mc.listRelatives(sel[0], s=True)[0]
            if mc.nodeType(shape) == 'nurbsCurve':
                target = sel[0]
            else:
                target = None
        elif mc.nodeType(sel[0]) == 'nurbsCurve':
            transform = mc.listRelatives(sel[0], p=True)[0]
            target = transform
    if target:
        mc.textFieldButtonGrp(TFBG_shapeLine, e=True, tx=target)
        mc.button(BT_shapeLine, e=True, en=True)
    else:
        mc.textFieldButtonGrp(TFBG_shapeLine, e=True, tx='')
        mc.button(BT_shapeLine, e=True, en=False)
    mc.refresh()
    addShapeCurve_popupMenu()


def addShapeCurve_popupMenu():
    if pf_73('addShapeCurve_popupMenu'):
        mc.deleteUI('addShapeCurve_popupMenu')
    target = mc.textFieldButtonGrp(TFBG_shapeLine, q=True, tx=True)
    if target and mc.objExists(target):
        mc.popupMenu('addShapeCurve_popupMenu', p=TFBG_shapeLine)
        mc.menuItem(l='Select Curve', c='%s.selShapeCurve()' % __name__)
        mc.menuItem(l='Remove Curve', c='%s.removeShapeCurve()' % __name__)
        mc.menuItem(d=True)
        mc.menuItem(l='Curve Visibility', cb=mc.getAttr(target + '.v'), c='%s.toggleShapeCurveVis()' %
__name__)
        mc.radioMenuItemCollection(p='addShapeCurve_popupMenu')
        mc.menuItem(l='Normal', rb=True if not mc.getAttr(target + '.overrideEnabled') or
mc.getAttr('%s.overrideDisplayType' % target) == 0 else False, c='%s.toggleShapeCurveOverride(mode=0)' %
__name__)
        mc.menuItem(l='Template', rb=True if mc.getAttr('%s.overrideDisplayType' % target) == 1 and
mc.getAttr(target + '.overrideEnabled') else False, c='%s.toggleShapeCurveOverride(mode=1)' % __name__)
        mc.menuItem(l='Reference', rb=True if mc.getAttr('%s.overrideDisplayType' % target) == 2 and
mc.getAttr(target + '.overrideEnabled') else False, c='%s.toggleShapeCurveOverride(mode=2)' % __name__)


def refreshShapeCurve():
    target = mc.textFieldButtonGrp(TFBG_shapeLine, q=True, tx=True)
```

```
    if not mc.objExists(target):
        removeShapeCurve()
        pf_71('%s not exists!' % target)


def selShapeCurve():
    target = mc.textFieldButtonGrp(TFBG_shapeLine, q=True, tx=True)
    if mc.objExists(target):
        mc.select(target)
    refreshShapeCurve()


def removeShapeCurve():
    mc.textFieldButtonGrp(TFBG_shapeLine, e=True, tx='')
    mc.button(BT_shapeLine, e=True, en=False)
    addShapeCurve_popupMenu()


def toggleShapeCurveVis():
    target = mc.textFieldButtonGrp(TFBG_shapeLine, q=True, tx=True)
    if mc.objExists(target):
        state = mc.getAttr(target + '.v')
        mc.setAttr(target + '.v', not state)
    refreshShapeCurve()


def toggleShapeCurveOverride(mode = 0):
    target = mc.textFieldButtonGrp(TFBG_shapeLine, q=True, tx=True)
    if mc.objExists(target):
        state = mc.getAttr(target + '.overrideEnabled')
        if not state:
            mc.setAttr(target + '.overrideEnabled', not state)
        mc.setAttr(target + '.overrideDisplayType', mode)
    refreshShapeCurve()


def spin(spinType = None, dialog = False, reverse = False):
    global firstID
    pf_61()
    comps = pf_06('face')
    if comps:
        selComps, selFaces, selMode = comps
    else:
        return False
    if not spinType:
        continua = pf_18(selFaces)
        if continua:
            outlineEdgesNum = pf_09(selFaces, num=True)
            if outlineEdgesNum % 2 == 0 and outlineEdgesNum >= 4:
                if pf_12(selFaces):
                    spinType = 'pf_52'
                elif len(selFaces) == 2 and len(pf_64(selFaces)) == 1:
                    spinType = 'pf_49'
                elif pf_11(selFaces):
                    spinType = 'pf_51'
                elif pf_13(selFaces):
                    spinType = 'pf_54'
```

```
            elif pf_14(selFaces):
                spinType = 'pf_53'
            elif pf_15(selFaces, selMode):
                spinType = 'pf_50'
        elif len(selFaces) == 2 and len(pf_64(selFaces)) == 1:
            spinType = 'pf_49'
    if spinType:
        delMeshHistory()
        messageDict = {'pf_50': 'Spin %s-loop:' % len(selFaces),
         'pf_52': 'Spin %s-ribbon:' % len(selFaces),
         'pf_51': 'Spin %s-star:' % len(selFaces),
         'pf_53': 'Spin %s-grid:' % pf_14(selFaces),
         'pf_49': 'Spin edge:',
         'pf_54': 'Spin H-shape:'}
        if spinType == 'pf_50':
            sortedVtxs_A, sortedVtxs_B = pf_05(selFaces, selMode)
            vtxNum = len(selFaces)
        if spinType in ('pf_53', 'pf_54'):
            mesh = pf_29()
            target = None
            if pf_73('Game_Poly_Tools_shrinkWrapUI') and mc.button(BT_shrinkWrap, q=True, en=True):
                target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
                if target:
                    if mc.objExists(target):
                        targetMesh = target
                    else:
                        target = None
            if not target:
                targetMesh = mc.duplicate(mesh)[0]
                mc.setAttr(targetMesh + '.v', False)
            if spinType == 'pf_53':
                gridType = pf_14(selFaces)
        if not dialog:
            pf_22()
            if spinType == 'pf_51':
                pf_51(selMode)
            elif spinType == 'pf_52':
                pf_52(selMode, reverse, 1)
            elif spinType == 'pf_49':
                pf_49(selMode, reverse, 1)
            elif spinType == 'pf_54':
                pf_54(selMode, reverse, mesh, targetMesh)
                if not target:
                    mc.delete(targetMesh)
            elif spinType == 'pf_53':
                pf_53(selMode, reverse, 1, gridType, mesh, targetMesh)
                if not target:
                    mc.delete(targetMesh)
            elif spinType == 'pf_50':
                pf_50(selMode, vtxNum, sortedVtxs_A, sortedVtxs_B, -1 if reverse else 1)
            pf_22()
            return 1
        firstID = 0
        opNum = 0
        if spinType in ('pf_49', 'pf_54'):
            bts = ['<--',
             'OK',
```

```
            'Cancel',
            '-->']
        elif spinType == 'pf_51':
            bts = ['OK', 'Cancel', '-->']
        elif spinType == 'pf_52' and len(selFaces) == 2:
            bts = ['<--',
             'OK',
             'Cancel',
             '-->']
        elif spinType == 'pf_53' and len(selFaces) == 6 or spinType == 'pf_50':
            bts = ['<<--',
             '<--',
             'OK',
             'Cancel',
             '-->',
             '-->>']
        else:
            bts = ['<<--',
             '<--',
             'OK',
             'Cancel',
             '-->',
             '-->>',
             '90 Degree']
        mc.undoInfo(ock=True)
        while True:
            msg = mc.confirmDialog(t='Spin - Game_Poly_Tools', m=messageDict[spinType], b=bts, db='OK',
ds='Cancel', cb='Cancel')
            pf_22()
            if msg == 'OK':
                if spinType in ('pf_53', 'pf_54') and not target and mc.objExists(targetMesh):
                    mc.delete(targetMesh)
                mc.undoInfo(cck=True)
                pf_22()
                delMeshHistory()
                return opNum
            if msg == 'Cancel':
                mc.undoInfo(cck=True)
                pf_22()
                if opNum:
                    pf_35()
                    if spinType in ('pf_53', 'pf_54') and not target and mc.objExists(targetMesh):
                        mc.delete(targetMesh)
                return 0
            if spinType == 'pf_50':
                if msg == '<<--':
                    for i in range(2):
                        firstID -= 1
                        pf_50(selMode, vtxNum, sortedVtxs_A, sortedVtxs_B, firstID)

                elif msg == '<--':
                    firstID -= 1
                    pf_50(selMode, vtxNum, sortedVtxs_A, sortedVtxs_B, firstID)
                elif msg == '-->':
                    firstID += 1
                    pf_50(selMode, vtxNum, sortedVtxs_A, sortedVtxs_B, firstID)
                elif msg == '-->>':
```

```
                for i in range(2):
                    firstID += 1
                    pf_50(selMode, vtxNum, sortedVtxs_A, sortedVtxs_B, firstID)

            elif spinType == 'pf_51':
                if msg == '-->':
                    pf_51(selMode)
            elif spinType == 'pf_52':
                if msg == '<<--':
                    pf_52(selMode, True, 2)
                elif msg == '<--':
                    pf_52(selMode, True, 1)
                elif msg == '-->':
                    pf_52(selMode, False, 1)
                elif msg == '-->>':
                    pf_52(selMode, False, 2)
                elif msg == '90 Degree':
                    pf_52(selMode, False, 3)
            elif spinType == 'pf_54':
                if msg == '<--':
                    pf_54(selMode, True, mesh, targetMesh)
                elif msg == '-->':
                    pf_54(selMode, False, mesh, targetMesh)
            elif spinType == 'pf_53':
                if msg == '<<--':
                    pf_53(selMode, True, 2, gridType, mesh, targetMesh)
                elif msg == '<--':
                    pf_53(selMode, True, 1, gridType, mesh, targetMesh)
                elif msg == '-->':
                    pf_53(selMode, False, 1, gridType, mesh, targetMesh)
                elif msg == '-->>':
                    pf_53(selMode, False, 2, gridType, mesh, targetMesh)
                elif msg == '90 Degree':
                    pf_53(selMode, False, 3, gridType, mesh, targetMesh)
            elif spinType == 'pf_49':
                if msg == '<<--':
                    pf_49(selMode, True, 2)
                elif msg == '<--':
                    pf_49(selMode, True, 1)
                elif msg == '-->':
                    pf_49(selMode, False, 1)
                elif msg == '-->>':
                    pf_49(selMode, False, 2)
                elif msg == '90 Degree':
                    pf_49(selMode, False, 3)
            mc.refresh()
            opNum += 1
    pf_48(targetMesh, mesh, 1, [pf_32(centerVtx)])
    if not target:
        mc.delete(targetMesh)
    mc.undoInfo(swf=True)
    newFaces = mc.polyListComponentConversion(centerVtx, fv=True, tf=True)
    mm.eval('doMenuComponentSelection("%s", "facet")' % mesh)
    mc.select(newFaces)


def pf_59(mesh):
```

```
    target = None
    if pf_73('Game_Poly_Tools_shrinkWrapUI'):
        target = mc.textFieldButtonGrp(TFBG_shrinkWrap, q=True, tx=True)
        if target:
            if mc.objExists(target):
                targetMesh = target
            else:
                target = None
    if not target:
        mc.undoInfo(swf=False)
        targetMesh = mc.duplicate(mesh)[0]
        mc.setAttr(targetMesh + '.v', False)
        mc.undoInfo(swf=True)
    mc.polyExtrudeFacet()
    mm.eval('polyMergeToCenter')
    centerVtx = mc.filterExpand(sm=31)[0]
    centerEdges = mc.filterExpand(mc.polyListComponentConversion(centerVtx, fv=True, te=True), sm=32)
    mm.eval('PolySelectConvert 1')
    mc.delete(centerEdges)
    delMeshHistory()
    polyFace = mc.filterExpand(sm=34)[0]
    sortedVtxs = pf_02(polyFace)
    sortedVtxsEdgeNums = [ len(mc.filterExpand(mc.polyListComponentConversion(vtx, fv=True, te=True),
sm=32)) for vtx in sortedVtxs ]
    startId = (sortedVtxsEdgeNums.index(min(sortedVtxsEdgeNums)) - 1) % 6
    edge_A = pf_24(sortedVtxs[startId], sortedVtxs[(startId + 2) % 6])
    edge_B = pf_24(sortedVtxs[(startId + 3) % 6], sortedVtxs[(startId + 5) % 6])
    newVtxs = pf_26(edge_A, edge_B)[:2]
    mc.undoInfo(swf=False)
    for i in range(5):
        mc.polyAverageVertex(newVtxs)
        delMeshHistory()

    pf_48(targetMesh, mesh, 2, [ pf_32(vtx) for vtx in newVtxs ])
    if not target:
        mc.delete(targetMesh)
    mc.undoInfo(swf=True)
    newFaces = mc.polyListComponentConversion(newVtxs, fv=True, tf=True)
    mm.eval('doMenuComponentSelection("%s", "facet")' % mesh)
    mc.select(newFaces)


def pf_60(toolbox, titleBarHeight = 21):
    state = mc.window(toolbox, q=True, tb=True)
    topEdge = mc.window(toolbox, q=True, te=True)
    leftEdge = mc.window(toolbox, q=True, le=True)
    teOffset = titleBarHeight
    height = mc.window(toolbox, q=True, h=True)
    if state:
        topEdge += teOffset
        leftEdge += 3
    else:
        topEdge -= teOffset
        leftEdge -= 3
    mc.window(toolbox, e=True, tb=not state, te=topEdge, le=leftEdge, h=height)
    return state
```

```
def pf_61():
    pass


def pf_62(smooth = True):
    allGrps = None
    selEdges = mc.filterExpand(sm=32)
    if selEdges and len(selEdges) >= 3:
        vtxGrps = pf_03(selEdges)
        if vtxGrps:
            allGrps = [ vtxsGrp for vtxsGrp in vtxGrps[0] + vtxGrps[1] if len(vtxsGrp) >= 3 ]
    else:
        selVtxs = mc.filterExpand(sm=31)
        if selVtxs and len(selVtxs) == 3:
            allGrps = [selVtxs]
        else:
            selFaces = mc.filterExpand(sm=34)
            if selFaces:
                outlineEdges = pf_09(selFaces)
                vtxGrps = pf_03(outlineEdges)
                allGrps = [ vtxsGrp for vtxsGrp in vtxGrps[1] if len(vtxsGrp) >= 3 ]
    if allGrps:
        for vtxsGrp in allGrps:
            vtxNum = len(vtxsGrp)
            vtxPosGroup = [ pf_69(mc.xform(vtx, q=True, ws=True, t=True)) for vtx in vtxsGrp ]
            temPoly = mc.polyCreateFacet(p=[ vec.toTuple() for vec in vtxPosGroup ])[0]
            mc.xform(temPoly, cp=True)
            centerPos = pf_69(mc.xform(temPoly, q=True, ws=True, rp=True))
            distGroup = [ vec.distance(centerPos) for vec in vtxPosGroup ]
            radius = sum(distGroup) / vtxNum
            circle = mc.circle(s=vtxNum, r=radius, c=(0, 0, 0), d=3 if smooth else 1, ch=False)[0]
            mc.xform(circle, ws=True, t=centerPos.toTuple())
            pos_axis_grp = [ abs(vec.x - centerPos.x) for vec in vtxPosGroup ]
            firstId = pos_axis_grp.index(min(pos_axis_grp))
            wuVec = (vtxPosGroup[firstId] - centerPos).toTuple()
            mc.delete(mc.normalConstraint(temPoly, circle, aim=[0, 0, 1], u=[0, 1, 0], wu=wuVec))
            mc.delete(temPoly)


def edge_pos(edge, vtx):
    if int(mc.polyInfo(edge, ev=True)[0].split()[2]) == pf_32(vtx):
        return 0
    else:
        return 1
```