# Планирование тестирования

Для полной проверки работоспособности программы для цифрового скульптинга необходимо применить такие виды тестирования как:

* Интеграционное тестирование;
* Функциональное тестирование;
* Тестирование удобства пользования (юзабилити);
* Тестирование производительности;
* Тестирование безопасности.

Тестировать разрабатываемый продукт необходимо разработчику. Все компоненты подвергнуты тестированию не будут, необходима проверка тех, которые угрожают наибольшими потерями для всего проекта, сосредоточенная на проверке того, что данный продукт должен выполнять.

## Интеграционное тестирование

Целью интеграционного тестирования является проверка совместной работы нескольких модулей разных api. Но так как я использую движок при создании данного программного средства, то все компоненты были протестированы ранее разработчиками данного программного обеспечения при его создании.

## Функциональное тестирование

### Процесс основанный на заранее известном (в некоторых

### случая предполагаемом) поведении пользователя, основанный в первую очередь на детальном анализе и изучении функциональной спецификации приложения, системы или небольшого модуля (компонента). Работа форм, нацеленных за сбор данных посетителей

Для работы с мешем используются кисти. Проверим, как будет вести себя программа при разных настройках кисти.

Сперва используем кисть, которая в диаметре будет больше, чем сам меш, и посмотрим, что их этого получится.

Результат представлен на рисунке 1.

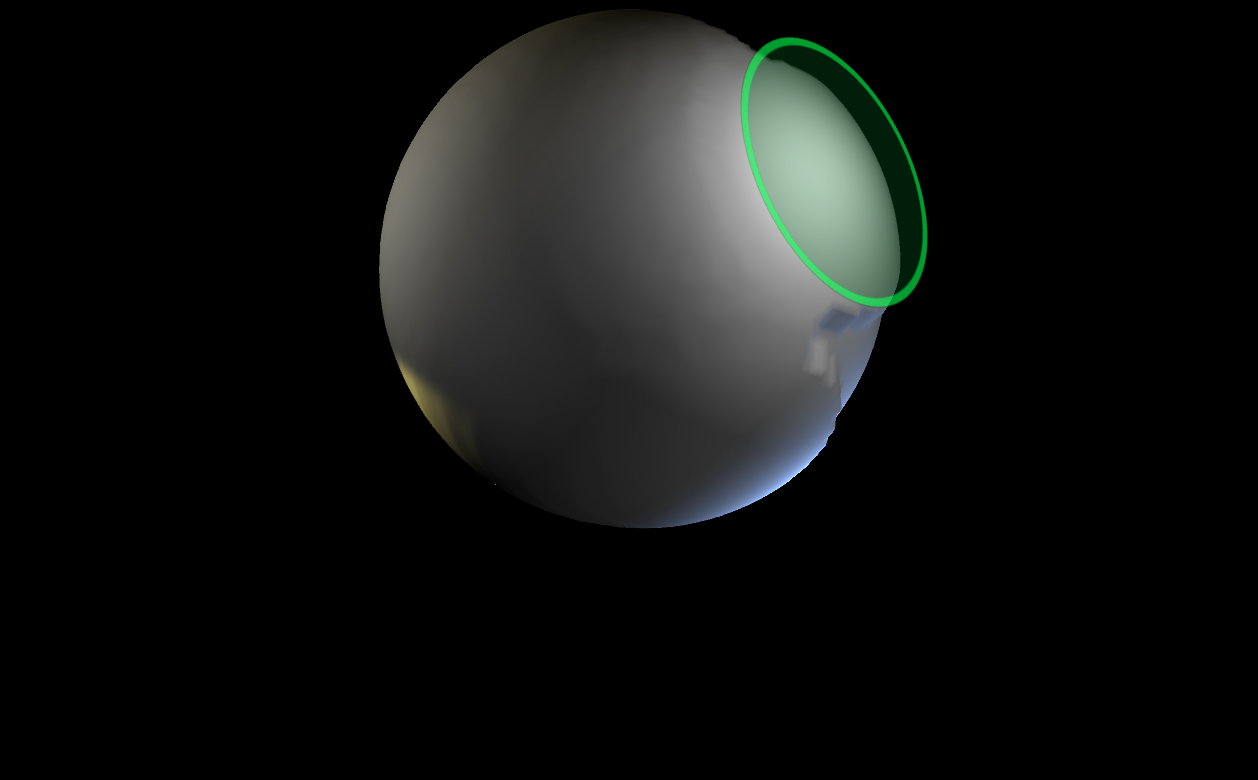


Рис 1 – Поведение программы.

Как видим, программа ведет себя адекватно, а следовательно – продолжает выполнять свои функции даже при дебильных настройках пользователя.

Проделаем тоже самое, только выставим диаметр кисти на минимальную величину. Результат на рисунке 2.



Рисунок 2 – поведение программы

В дальнейшем планируется описать алгоритм тессаляции, дабы была возможность увеличивать к-во полигонов на меше, делая его при этом более гладким, и детализированным. Возможные проблемы: пользователь будет тессалировать меш, и у него случится нехватка оперативной памяти. Нужно будет выдавать ему сообщение о том, что у него заканчивается оперативная память, либо попросту установить некий лимит на количество уровней тессаляции.

1. Тестирование производительности

На данный момент тест производительности невозможен, так как еще не реализован алгоритм тессаляции. Это единственный способ нагрузить систему, и проверить ее работоспособность.

1. Тестирование безопасности

Проводить тестирование безопасности для данного продукта не имеет смысла, так как программа не имеет доступа к сети Интернет, и не содержит каких-либо важных данных, которые могут быть похищены в ходе атаки на данное программное обеспечение.

1. Тестирование удобства

Тестирование удобства пользования - это метод тестирования, направленный на установление степени удобства использования, обучаемости, понятности и привлекательности для пользователей разрабатываемого продукта в контексте заданных условий.

Тестирование удобства пользования дает оценку уровня удобства использования приложения по следующим пунктам:

производительность, эффективность (efficiency) - сколько времени и шагов понадобится пользователю для завершения основных задач приложения, например, размещение новости, регистрации, покупка и т.д.? (меньше - лучше)

правильность (accuracy) - сколько ошибок сделал пользователь во время работы с приложением? (меньше - лучше)

активизация в памяти (recall) – как много пользователь помнит о работе приложения после приостановки работы с ним на длительный период времени? (повторное выполнение операций после перерыва должно проходить быстрее чем у нового пользователя)

эмоциональная реакция (emotional response) – как пользователь себя чувствует после завершения задачи - растерян, испытал стресс? Порекомендует ли пользователь систему своим друзьям? (положительная реакция - лучше)

Тестирование удобства для данного программного обеспечения провалено, в данный момент ведутся работы для улучшения интерфейса.

1. Тестирование

Для частичной проверки работоспособности системы приложение были выбраны модули SculpterController и Sculpter.

Процесс тестирования можно разбить на 5 атрибутов:

- Уникальный идентификатор тест кейса

- Название

- Предусловие

- Шаги

- Ожидаемый результат

Рассмотрим данные атрибуты для каждого модуля.

Модуль SculpterController

Данный тест-кейс имеет идентификатор 01.

Название тест-кейса : «Тестирование поведения настройщика кистей».

Краткое описание.

Данный тест-кейс направлен на выявление ошибок при настройках кистей.

Для данного кейса предусловие отсутсвует.

Основные этапы для получения результата:

1. Выбрать кисть
2. Сместить ползунок настроек.

Ожидаемый результат представлен в таблице 1.

Модуль Sculpter.

Данный тест-кейс имеет идентификатор 02.

Название тест-кейса: «Проверка поведения кисти при наведении в зону меша, и в зону вне меша».

Для данного кейса предусловие отсутствует.

Основные действия для получения результата:

1. Запустить программу
2. Навести кисть на меш.

Ожидаемый результат представлен в таблице 1.

Таблица 1

|  |  |  |
| --- | --- | --- |
| Переменные | Присвоенное значение | Ожидаемый результат |
| brushRad  BrushHeight | 100  100 | Кисть будет работать в штатном режиме. Если же пользователю каким-то неизвестным образом удастся вытянуть ползунок на значение, которое больше 100, то будет выведено сообщение: «Вы, видимо, тетка с бухгалтерии» |
| Color | Green | При наведении на меш, кисть будет принимать зеленый цвет, если наведена не на меш, а на пустое место в сцене, то кисть примет белый цвет. |