УДК 004

РАЗРАБОТКА ПЛАГИНА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ BLENDER

Кряжева Е. В.,

к.псих.н., доцент,

Калужский государственный университет им. К.Э. Циолковского,

Калуга, Россия

Овсепян А.М.,

магистрант,

Калужский государственный университет им. К.Э. Циолковского,

Калуга, Россия

Аннотация.

В статье рассматривается процесс разработки плагина программного обеспечения blender. Описывается программа blender и ее функциональные возможности. Анализируются способы расширения его функциональности через создание add-on. Обосновывается использование Python API для разработки плагина и пример скрипта. Предлагается пример add-on, который выполняет функцию создания фигуры (mesh) трубки, и его программный код. Описывается процесс регистрации скрипта в blender и процесс создания пользовательского интерфейса под конкретную задачу. В конце представлены выводы по проделанной работе и дальнейшие перспективы расширения возможностей blender за счет разработки плагинов.

Ключевые слова: blender, плагин, add-on, скрипт, программный код, пользовательский интерфейс, python, python API, функция, надстройка.

BLENDER SOFTWARE PLUGIN DEVELOPMENT

Kryazheva E. V.,

Candidate of Psychological Sciences, Associate Professor,

Kaluga State University named after K.E. Tsiolkovsky,
Kaluga, Russia
Hovsepyan A.M.,
Undergraduate,
Kaluga State University named after K.E. Tsiolkovsky,
Kaluga, Russia

Annotation.
The article discusses the process of developing a blender software plugin. The blender program and its functionality are described. Ways to extend its functionality through the creation of an add-on are analyzed. The use of the Python API for plugin development and an example script are justified. An example of an add-on, which performs the function of creating a mesh of a tube, and its program code are presented. The process of registering a script in blender and the process of creating a user interface for a specific task are described. At the end, the conclusions on the work done and further prospects for expanding the capabilities of blender through the development of plugins are presented.
Keywords: blender, plugin, add-on, script, program code, user interface, python, python API, function, add-on.

Blender представляет собой программу для трёхмерного моделирования объектов. Является программным обеспечением с открытым исходным кодом, и включает в себя следующие средства: моделирование, рендеринг, анимацию и монтаж, редактирование видео, визуальные эффекты, композитинг, текстурирование и многие типы симуляций.

Одним из главных преимуществ Blender является его способность расширять функциональность через создание add-on. Add-on - это небольшая программа, которая добавляет новые инструменты и возможности в Blender,

позволяя пользователю настроить программу под свои потребности. В этой статье мы рассмотрим процесс создания такого add-on для Blender.

Данная программа отличается большой универсальностью и самодостаточностью, поскольку содержит практически исчерпывающий набор программных инструментов, необходимых для обеспечения всей технологической цепочки динамической компьютерной визуализации любого уровня сложности.

Перед тем, как начать создавать add-on, необходимо установить Blender и настроить среду разработки. Для этого понадобится язык программирования Python, так как Blender использует Python API для создания add-on. Python API (Application Programming Interface) — это набор функций и процедур, предоставляемых конкретной библиотекой или фреймворком на языке Python, для взаимодействия с другими программами или сервисами. API определяет интерфейс для взаимодействия между различными компонентами программного обеспечения, позволяя им обмениваться данными или вызывать функции друг у друга.

Для начала создания add-on понадобится файл Python с расширением .py. Необходимо создать новый файл и назвать его так, как будет называться add-on. Например, можно назвать его "my_first_addon.py". Add-on будет находиться в виде скрипта Python. Нужно открыть файл my_first_addon.py в любой программе для редактирования текста и начать добавлять код.

Сначала необходимо прописать скрипт создания трубы. Для превращения скрипта в аддон, необходимо скрипт обернуть в класс. API требует, чтобы классы обязательно наследовались от нескольких преопределенных классов:

bpy.types.Panel

bpy.types.Menu

bpy.types.Operator

bpy.types.PropertyGroup

bpy.types.KeyingSet

bpy.types.RenderEngine

API так же требует, чтобы пользовательские классы были статическими, следовательно в них нет необходимости определять конструктор _init_ и деструктор _del_. Класс bpy.types.operator есть функция execute, которая выполняется в момент обращения к классу через API Blender. Оборачиваем код скрипта в класс и наследуем его от bpy.types.operator. После переопределения функции execute и занесения в нее исполняемый код нашего скрипта, функция execute возвратит указание об успешном выполнении {‘FINISHED’}.

- bpy.context - Справочник по API. Удобно иметь список доступных элементов, с которыми может работать ваш скрипт.
- bpy.types.Operator– Следующие дополнения определяют операторов, в этих документах приводятся подробности и дополнительные примеры операторов.
- bl_info — это словарь, содержащий метаданные надстройки, такие как название, версия и автор, которые будут отображаться в списке надстроек. Он также определяет минимальную версию Blender, необходимую для запуска скрипта; более старые версии не будут отображать надстройку в списке.
- register — это функция, которая запускается только при включении надстройки. Это означает, что модуль можно загрузить без активации надстройки.
- unregister — это функция для выгрузки любых настроек register, она вызывается, когда надстройка отключена.

Надстройка обычно регистрирует операторов, панели, пункты меню и т.д., но стоит отметить, что это может сделать любой скрипт, запущенный из текстового редактора или даже из интерактивной консоли. В надстройке нет

ничего особенного, что позволяет ему интегрироваться с Blender, такая функциональность предоставляется модулем blender_api: bpy.

Таким образом, надстройка — это всего лишь способ инкапсулировать (скрыть детали реализации и оставить только визуальный результат) модуль Python таким образом, чтобы пользователь мог легко его использовать. Пример add-on, который выполняет функцию создания фигуры (mesh) трубки.

```
import math
import bpy
class createTube(bpy.types.Operator):
   def execute(self, context):
      n = 8
      r1 = 1
      r2 = 0.5
      h = 1

      verts = []
      ring1 = []
      ring2 = []
      ring3 = []
      ring4 = []
      faces = []

      for i in range(n):
         grad = (360 * i / n) * math.pi / 180
         ring1.append([r1 * math.cos(grad), r1 * math.sin(grad), 0])
         ring2.append([r2 * math.cos(grad), r2 * math.sin(grad), 0])
         ring3.append([r1 * math.cos(grad), r1 * math.sin(grad), h])
         ring4.append([r2 * math.cos(grad), r2 * math.sin(grad), h])
         if i == 0:
            faces.append([i - 1 + n, i, i + 2 * n, i - 1 + 3 * n])
            faces.append([i - 1 + 2 * n, i - 1 + 4 * n, i + 3 * n, i + n])
            faces.append([i - 1 + n, i - 1 + 2 * n, i + n, i])
            faces.append([i - 1 + 3 * n, i + 2 * n, i + 3 * n, i - 1 + 4 * n])
         else:
            faces.append([i - 1, i, i + 2 * n, i - 1 + 2 * n])
            faces.append([i - 1 + n, i - 1 + 3 * n, i + 3 * n, i + n])
            faces.append([i - 1, i - 1 + n, i + n, i])
            faces.append([i - 1 + 2 * n, i + 2 * n, i + 3 * n, i - 1 + 3 * n])

      verts.extend(ring1)
      verts.extend(ring2)
      verts.extend(ring3)
      verts.extend(ring4)

      tubeMesh = bpy.data.meshes.new("Tube")
      tubeMesh.from_pydata(verts, [], faces)
      tubeMesh.update()
      tube = bpy.data.objects.new("Tube", tubeMesh)
      bpy.context.scene.objects.link(tube)
```

```
bpy.ops.object.select_all(action="DESELECT")
tube.select = True
bpy.context.scene.objects.active = tube
return {'FINISHED'}
```

Так, класс наследующий bpy.types.operator становится оператором Blender API.

В начале скрипта необходимо зарегистрировать add-on в Blender. Для этого используется функция register () и добавляются обработчики событий и настройки создаваемого add-on.

Любой класс, зарегистрированный в Blender API, должен иметь уникальный идентификатор, чтобы можно было к нему обращаться по этому идентификатору. Для определения идентификатора нашего класса создадим такую переменную и присвоим ей значение ‘mesh.create_tube’:

bl_idname = 'mesh.create_tube'

Для окончательного оформления плагина необходимо создать его описание. Словарь с предопределенным именем bl_info для составления описания плагина, имеет следующие предопределенные пункты:

1. name — название плагина
2. author — ФИО автора
3. version — версия плагина
4. blender — версия Blender под которую разрабатывался плагин
5. category — категория, в которую плагин будет помещен
6. location — указание на то, где искать панель плагина
7. url — указание на исходный код плагина (откуда он распространяется)
8. description — строка с более подробным описанием плагина

Нет необходимости заполнять все пункты, однако некоторые из них являются важными.

По минимуму стоит заполнять пункты name, category и blender.

```
bl_info = {
   'name': my_first_addon',
```

```
    'author': 'incklip',
    'version': (0, 0, 1),
    'blender': (4, 0, 0),
    'category': 'Add Mesh',
    'description': 'This addon adds tor mesh to scene'
}
if __name__ == "__main__" :
    register()
```

Этот код нужен исключительно в процессе разработки аддона. Если код аддона или ссылка на него в IDE набраны в окне Text Editor – при наличии этого участка кода по нажатию на кнопку Run Script будет выполняться активация аддона в Blender, что позволит сразу протестировать его работу. В завершенном аддоне этот код совершенно не нужен.

Следующий шаг - создание пользовательского интерфейса (UI) для вашего add-on. Blender предоставляет различные виджеты и элементы управления для создания UI. Необходимо добавить элементы, такие как кнопки, поля ввода и выпадающие списки для создаваемого add-on.

Далее необходимо добавить в add-on обработчики. Важно определение реагирования add-on на действия пользователя, такие как нажатия кнопок или изменение значений в полях ввода. Нужно написать функции, которые будут вызываться при наступлении этих событий и определить требуемое поведение создаваемого add-on.

Панель инструментов - это место, где будут отображаться и доступны функции нами создаваемого add-on. Необходимо зарегистрировать панель инструментов с использованием функции bpy.utils.register_class(). После нужно добавить новую панель инструментов в разделе, где мы хотим разместить наш add-on. Есть возможность выбрать существующий раздел или создать новый.

Сохраняем наш скрипт add-on и переходим в Blender. Зайдем в настройки Blender и в разделе "Add-ons" найдем нами созданный add-on. Далее активируем его и используем по назначению. После того, как нами созданный add-on работает без ошибок и готов к использованию, мы можете поделиться

им с другими пользователями Blender. Мы можем опубликовать наш add-on на платформах для обмена add-on, таких как Blender Market или GitHub.

Таким образом, создание add-on для программного обеспечения Blender дает возможность расширить функциональность программы и настроить ее под свои потребности. Начиная с подготовки и настройки среды разработки, процесс включает в себя создание основы, добавление пользовательского интерфейса, обработку событий и регистрацию панели инструментов. После тестирования и установки вашего add-on продуктом можно поделиться с другими пользователями Blender.

Библиографический список:

1. Add-on Tutorial // docs.blender.org URL: https://docs.blender.org/manual/en/latest/advanced/scripting/addon_tutorial.html (дата обращения: 12.01.2024).
2. Add-ons // docs.blender.org URL: https:// docs.blender.org/manual/en/latest/addons/index.html (дата обращения: 15.01.2024).
3. Python 3.x в контексте 3D-редактора Blender. [Электронный ресурс]. URL: http://blender3dlove.blogspot.com/2013/05/1-python-blender.html (дата обращения: 29.10.2018).
4. Blender 4.0 Python API Documentation // docs.blender.org URL: https://docs.blender.org/api/current/ (дата обращения: 10.01.2024).
5. Создание аддона для Blender // b3d.interplanety.org URL: https://b3d.interplanety.org/ sozdanie-addona-dlya-blender/ (дата обращения: 15.01.2024).

Оригинальность 86%