СРЕДСТВА ДЛЯ СОЗДАНИЯ РАССЧЕТНЫХ ВЕБ-ПРИЛОЖЕНИЙ НА PYTHON

СУТЧЕНКОВ А. А., ТИХОНОВ А. И.

Национальный исследовательский университет «МЭИ»
Москва, Российская Федерация
E-mail: SutchenkovAA@mpei.ru, tikhonovai@mpei.ru

В докладе рассмотрены средства для оперативной разработки и публикации небольших расчетных приложений — апплетов. Работать с апплетами можно как локально, так и через Интернет. Рассматривается технология создания апплетов. Средства позволяют с умеренной трудоемкостью публиковать интегрированные образовательные ресурсы, включающие в себя текст, графику, видео и апплеты. Приводятся примеры апплетов.

Ключевые слова: расчетные веб-приложения, оперативное создание веб-приложений, апплеты, инженерное образование.

ВВЕДЕНИЕ

Важное место в инженерном образовании занимают расчеты. Очень часто преподавателю для поддержки практических занятий, расчетных заданий, курсового проектирования, необходимы приложения, которые могли бы поддерживать данные виды занятий, проводить расчеты, обрабатывать экспериментальные данные, осуществлять представление результатов в простой и наглядной форме. При этом разработка приложений желательно должна осуществляться при подготовке к занятиям и не предъявлять к разработчику чрезмерных требований к его квалификации.

Как показывает опыт такие приложения небольшие и достаточно простые. В дальнейшем такие веб-приложения мы будем называть *апплетами*. Данный термин был введен в середине 90-х годов прошлого века разработчиками системы программирования Javaдля приложений, выполняемых в контексте браузера. Здесь под апплетами понимаются приложения, выполняющиеся как на стороне сервера, так и клиента.

Апплеты применяются не только для поддержки учебного процесса, но и в практической деятельности, так апплеты широко применяются в энергосбережении [1], где они представляют собой компьютерные реализации различных методик, в том числе и стандартов.

Для учебного процесса важным моментом является простота эксплуатации апплетов. Установка приложений перед проведением занятий в университетских компьютерных классах отнимает много времени и нервов, поэтому глубокое убеждение авторов состоит в том, что большинство приложений, используемых в учебном процессе, должны быть веб-приложениями. Веб-приложения не требуют установки на клиентском компьютере, для работы с

ними достаточно наличие браузера, они практически не зависят от вычислительного устройства и операционной системы на стороне пользователя. Недостатками веб-приложений является необходимость подключения к Интернету или корпоративной сети, а также сравнительная бедность пользовательских интерфейсов по сравнению с настольными приложениями.

В работе рассматривается среда, которая позволяет разрабатывать, публиковать и выполнять апплеты, кроме того, среда позволяет просто и оперативно публиковать интегрированные электронные ресурсы, включающие в себя html-, pdf-файлы, видео, апплеты.

ТЕХНОЛОГИЧЕСКАЯ ПЛАТФОРМА

В качестве технологической платформы для среды разработки апплетов были использованы Python 2.7 [2] с предустановленными пакетами numpy, scipy [3] для проведения научно-технических и инженерных расчетов, matplotlib [4] для отображения графики и веб-фреймворк web2py [5]. Python является интерпретируемым языком программирования высокого уровня с динамической типизацией и высоким быстродействием. Его отличительной особенностью является активное сообщество разработчиков приложений и, что самое важное, наличие огромного числа бесплатных библиотек для обработки и визуализации данных, проведения научно-технических расчетов, решения прикладных задач. Эти библиотеки достаточно качественные, их критические по быстродействию фрагменты написаны на компилируемых языках. В Интернете можно легко найти информацию по применению библиотек, задать и получить ответы на вопросы на профильных форумах.

Среда разработки апплетов сделана переносимой, т.е. в локальном варианте она не нуждается в установке и может запускаться с внешнего жесткого диска или даже флеш-накопителя. Данное решение принято для того, чтобы максимально облегчить разработку и применение апплетов, т.к. для разворачивания системы на компьютере достаточно просто скопировать на свой переносимый носитель каталог с системой. Среда разработки апплетов включает в себя только свободно распространяемые компоненты: Portable Python 2.5.7.1, numpy, scipy, matplotlib. Выбор веб-фреймворка web2py основан на сравнительной простоте применения, компактности, встроенном веб-сервере, простоте переноса на сервер как Windows, так и Linux.

Для работы с апплетами пользователю достаточно запустить командный файл на своем компьютере. Апплеты работают во всех современных браузерах, включая Chrome, Firefox, Opera, Safari, Internet Explorer, начиная с версии 9. Пользовательский интерфейс подстраивается под размеры экрана устройства пользователя.

ВЗАИМОДЕЙСТВИЕ ПОЛЬЗОВАТЕЛЯ С АППЛЕТОМ

С точки зрения пользователя взаимодействие с апплетом заключается в вводе входных данных и нажатии после этого кнопки **Передать**. Введенные данные передаются среде выполнения апплетов, где на их основе рассчитываются значения выходных переменных, строятся графики, результаты передаются в браузер пользователя и представляются пользователю в html-форме. На рис. 1 представлен апплет для моделирования броуновского движения.

Рис. 1. —Моделирование 100 шагов броуновского движения 10 частиц, траектории частиц отображаются цветом

При разработке апплета можно при желании опубликовать его описание и исходный код, которые отображаются в отдельном окне после нажатия соответствующей кнопки в верхней части окна апплета. На рис. 2 представлен фрагмент исходного кода апплета моделирования броуновского движения.

При отображении исходного текста осуществляется нумерация строк, расцвечивание синтаксических конструкций.

Исходный текст апплета 'Brown'

```
1.  #-------------------------------------------------------------------------------
2.  # Name:        Brown
3.  # Purpose:     Броуновское движение произвольного числа частиц
4.  #              в единичном квадрате
5.  # Author:      А. Тихонов
6.  #
7.  # Created:     20.08.13
8.  # Modified:    20.08.03
9.  # Copyright:   (c) А. Тихонов,  2013
10. # Licence:     MIT
11. #-------------------------------------------------------------------------------
12. #!/usr/bin/env python
13.
14. from appletsMain import * # это единственный обязательный импорт
15.
16. from cyrfont import cyrfont # для вывода кириллицы на графиках
17. import numpy as np
18.
19.
20.
21. class Brown(Applet):
22.     u'''
23.     Броуновское движение  произвольного числа частиц в двухмерном случае
24.     '''
25.
26.     def __init__(self):
27.         super(Brown, self).__init__()
28.
29.         dsc(self,u'Brown',   # короткое имя для отображения в пользовательском интерфейсе
30.             u'Броуновское движение',    # краткое описание для отображения в пользовательском интер
    фейсе
31.             author=u'А. Тихонов',    # сведения об авторе (авторах) апплета для отображения в польз
    овательском интерфейсе
32.             org=u'НИУ МЭИ',         # организация, где разработан апплет
33.             source=u'src/Brown.py',  # ссылка на исходный текст апплета
34.                                      # ссылка относительна каталога /applets/static/
35.                                      # данный параметр не является обязательным
36.             link=u'src/Brown.htm' # ссылка на описаниие апплета в формате html
37.                                      # ссылка относительна каталога /applets/static/
38.                                      # данный параметр не является обязательным
39.
40.             )
41.
```

Рис.2. — Фрагмент исходного кода апплета

ПУБЛИКАЦИЯ АППЛЕТОВ И ЭЛЕКТРОННЫХ РЕСУРСОВ

В среду разработки апплетов встроены средства публикации электронных ресурсов, включая html-, pdf-файлы, файлы исходных текстов, видео, апплеты. Кроме того, имеется возможность простой публикации интегрированных ресурсов, когда в htmlфайл встраиваются все перечисленные виды содержимого.

В качестве основной технологии публикации ресурсов предлагается следующее. Текст, формулы, изображения подготавливаются в текстовом редакторе, например, Microsoft Word, затем сохраняются в виде html-файлов. Встраивание активного содержимого в электронный ресурс (рис.3) осуществляется с помощью специальных тегов непосредственно в тексте ресурса, внутри тега нужно указать относительный указатель встраиваемого ресурса.

Апплеты и другие элементы электронных ресурсов организуются с помощью оглавления, расположенного в левой части окна веб-приложения апплетов (рис.4).

```
Встраивание апплета:
{{applet}}Trm{{/applet}}
Встраивание видео:
{{video}}src/Trm.mp4{{/video}}
Встраивание файлов PDF:
{{pdf}}src/Trm.pdf{{/pdf}}
Встраивание исходного кода:
{{code}}src/Trm.py{{/code}}
```

Рис.3. — Встраивание апплетов, видео, pdf-фалов, исходного кода

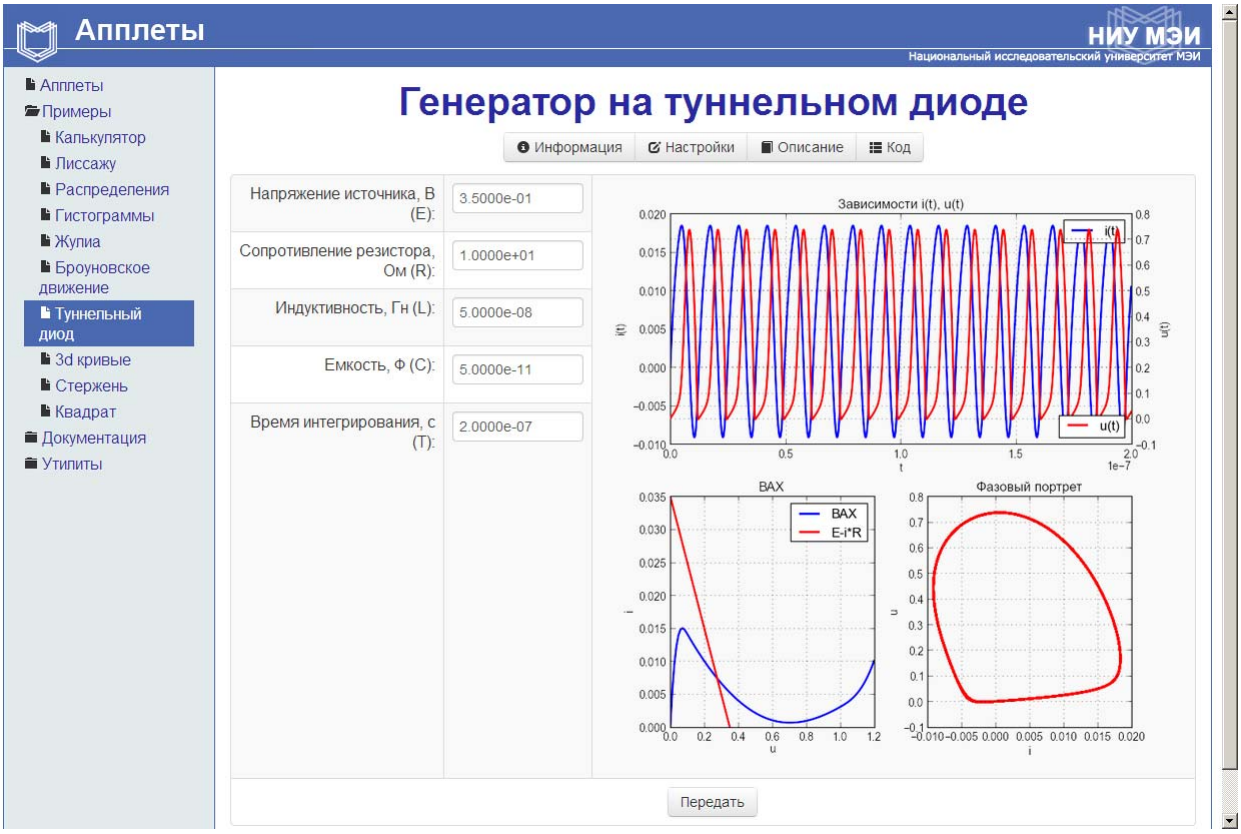

Рис.4. — Оглавление веб-приложения апплетов

Содержание веб-приложения апплетов отображается в виде дерева в левой части окна, см. рис. 4, В настоящее время оглавление подготавливается вручную в виде текстового файла toc.yaml. На рис. 5 представлен фрагмент оглавления.

```
#######################################################
#              Оглавление сайта                       #
#######################################################
-
    title: Апплеты
    dsc:   Инфраструктура апплетов
roles:  '*'
    url:    '/'
    notitle: false
-
    title: Примеры
    dsc:   Примерыапплетов
roles:  '*'
    url:    /local_content/examples
    notitle: false
    children:
        -
            title: Калькулятор
            dsc:   Простейшийкалькулятор
            roles: '*'
            url:   /app/calc
notitle: false
-
title: Лиссажу
            dsc:   Фигуры Лиссажу (графики
параметрических кривых)
roles: '*'
            url:   /app/Liss
notitle: false
```

Рис. 5. —Фрагмент оглавления веб-приложения апплетов

Синтаксис языка yaml, используемого для описания оглавления, близок к синтаксису Python.Кроме того, в Python имеются средства для анализа структур данных на этом языке. Обязательными атрибутами элемента оглавления является следующие:

– title —имя апплета, отображаемое в оглавлении,

– dsc — краткое описание, отображаемое при наведении курсора мыши на элемент оглавления;

– roles — группы пользователей, которым разрешен доступ к элементу,

– url — указатель ресурса элемента,

– notitle — равен false, если заголовок берется из текста описания элемента, если же задать true, то в качестве заголовка в панели содержимого в правой части окна приложения используется значение dsc,

– children — если данный элемент имеет дочерние элементы, то они перечисляются в данном элементе.

После внесения изменений в оглавление необходимо либо перезапустить веб-приложение апплетов, либо, что гораздо проще, выполнить обновление оглавления с помощью апплета, находящегося в разделе **Утилиты**.

ПРОЦЕСС РАЗРАБОТКИ АППЛЕТОВ

С точки зрения разработчика процесс создания апплета достаточно прост и включает в себя несколько шагов. Чтобы сделать его более наглядным, было снято видео процесса разработки апплета (раздел **Документация / Пример разработки апплетов**).

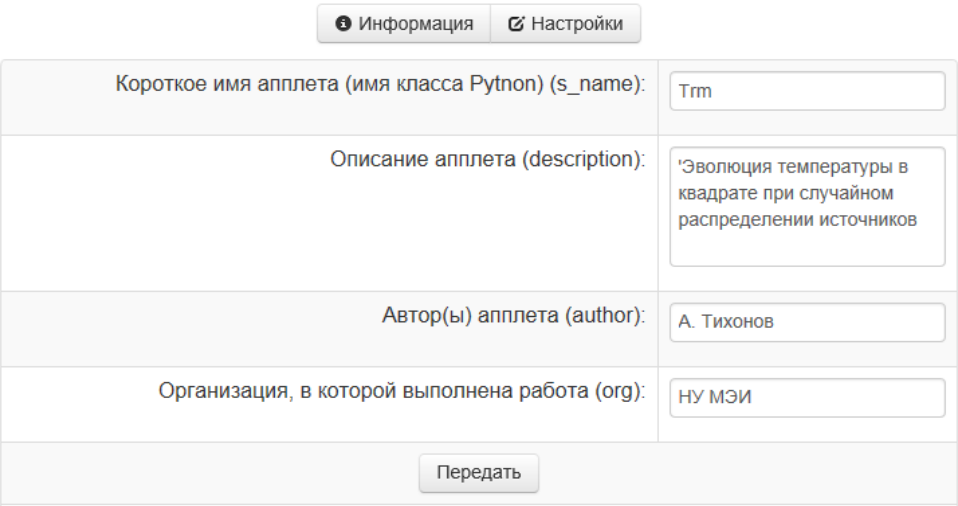

Рис. 6. — Утилита создания заготовки исходного кода апплета

Первым шагом является создание заготовки апплета, для чего разработчик должен выполнить утилиту **Конструктор** (рис. 7)

Сам апплет представляет собой класс, наследующий от класса Applet.

Вторым шагом является заполнение полей описания апплета в инициализаторе —методе __init__() (рис.7). Часть полей заполняются при создании заготовки апплета.

```
dsc(self,u'Trm',   # короткое имя
    u'Эволюция температуры в квадрате при случайном распределении
источников', # краткое описание
    author=u'А.Тихонов',   # сведения об авторе (авторах)
    org=u'НИУ МЭИ',        # организация, где разработан апплет
   source=u'src/Trm.py',  # ссылка на исходный текст апплета
    link=u'src/Trm.htm' # ссылка на описание апплета
)
```

Рис. 7. — Описание апплета

Описание апплета включает следующие обязательные поля: короткое имя, описание апплета — эти поля являются позиционными, в необязательном име-

нованном поле author указываются авторы, а поле org — организация, где выполнена разработка апплета.

Необязательные именованные поля source и link служат для задания указателей на ресурсы исходного кода и описания апплета, соответственно. Указатели апплета кодируются относительно каталога applets/static/texts/. Данную информацию (рис. 8) пользователь может увидеть, нажав кнопку **Информация**, расположенную в верхней части окна апплета (рис. 4). Информация об апплете отображается в отдельной вкладке браузера.

Информация об апплете

Апплет:	Trm
Описание апплета:	Эволюция температуры в квадрате при случайном распределении источников
Автор:	А.Тихонов
Организация:	НИУ МЭИ
Ссылка на описание апплета:	src/Trm.htm
Ссылка на исходный текст апплета:	src/Trm.py

Рис. 8. — Стандартная информация об апплете

Третьим шагом является описание всех интерфейсных переменных апплета. Под интерфейсными переменными, понимаются переменные, которые отображаются в интерфейсе пользователя. По описанию автоматически генерируется пользовательский интерфейс апплета. В свою очередь интерфейсные переменные становятся доступными в методах класса.

Описание интерфейсных переменных также осуществляется в методе __init__() апплета. Фрагмент описания интерфейсных переменных апплета представлен на рис. 9.

Объект описания интерфейсной переменной представляет собой экземпляр класса var. Интерфейсные переменные могут быть следующих типов: строка (Str), текст (Text)— отличается от строки тем, что отображается пользователю в текстовой области (textarea), а не текстовом поле, что удобнее для работы с длинными строками.

Численные данные могут быть представлены как целыми (Int), так и числами с плавающей точкой. Логические переменные (Bool) отображаются в пользовательском интерфейсе флажками.

Если необходимо дать пользователю возможность выбора из заданного набора альтернатив, то используются переменные типа Choice. В этом случае альтернативы отображаются пользователю в открывающемся списке.

```
var.Float(self,'L',u'Теплоемкость', value=1.0, defaultValue=1.0,
restrictions=[0.0001, 10000.0])

var.Float(self,'D',u'Теплопроводность', value=1.0,
defaultValue=1.0, restrictions=[0.0001, 10000.0])

var.Float(self,'P',u'Суммарнаямощность', value=1.0,
defaultValue=1.0, restrictions=[0.0001, 10000.0])

var.Int(self,'n_r',u'Числослучайнораспределенныхисточников
тепла', value=10, defaultValue=10, restrictions=[1,100])

var.Choice(self,'palette',u'Цветоваяпалитра', value='spectral',
defaultValue='spectral', inout='in',
restrictions=('gnuplot','gnuplot2', 'ocean', 'afmhot',
'rainbow', 'seismic'))

var.Bool(self,'success', u'Успешноевыполнениеапплета',
inout='out')

var.Picture(self,'picture1', u'картинка')
```

Рис. 9. — Фрагмент описания интерфейсных переменных

Пожалуй, наиболее интересным типом данных веб-приложения апплетов является пользовательский объект (UserObj), представляющий собой экземпляр класса, инициализируемый строкой, кроме того, для него должен быть определен метод __unicode__(), результат вызова которого отображается пользователю. Пользовательский объект может быть использован для различных нужд, например, для выдачи пользователю рандомизированных заданий.

Наконец, переменная типа Picture служит для отображения в интерфейсе пользователя картинок, рисуемых средствами библиотеки matplotlib.

Для каждой переменной необходимо определить имя переменной (оно должно соответствовать правилам определения имен Python) и описание переменной, которое может быть произвольной строкой. Имя и описание переменной является первыми двумя позиционными параметрами описания переменной.

Каждой переменной соответствует метод доступа к ней, который задается в именованном параметре inout. По отношению к методу доступа переменная может входной ('in'), входной/выходной ('inout'), это значение является умолчанием, выходной ('out') или скрытой 'hidden'. Переменные с методом доступа 'hidden' не отображаются пользователю и могут служить разработчику средством передачи данных между вызовами апплета пользователем.

Автоматически проверяемые веб-приложением ограничения на значения переменной определяются в параметре restriction. Если задать ограничение в виде списка, например, [1.0,2.0], то апплет не допустит ввода пользователем значений, меньших 1.0 и больших 2.0. Если ограничение задается набором зна-

чений, то оно кодируется кортежем, например, именно так задано ограничение на значение имени палитры на рис. 9. Напомним, что в этом случае пользователь выбирает значение интерфейсной переменной из раскрывающего списка.

Таким образом, при работе пользователя с апплетами осуществляется проверка введенных данных. Если необходимо провести более сложные проверки сочетаний значений переменных, то разработчик должен сделать это в методе апплета check().

Текущее значение интерфейсной переменной задается в параметре value, оно обязательно должно соответствовать типу переменной и удовлетворять ограничениям. При первом запуске апплета и в том случае, если не задано значение параметра value, используется умалчиваемое значение (параметр defaultValue). При определении интерфейсных переменных рекомендуется задать значение параметров value и defaultValue.

На рис. 10 представлен пользовательский интерфейс апплета Trm, сгенерированный по описаниям переменных.

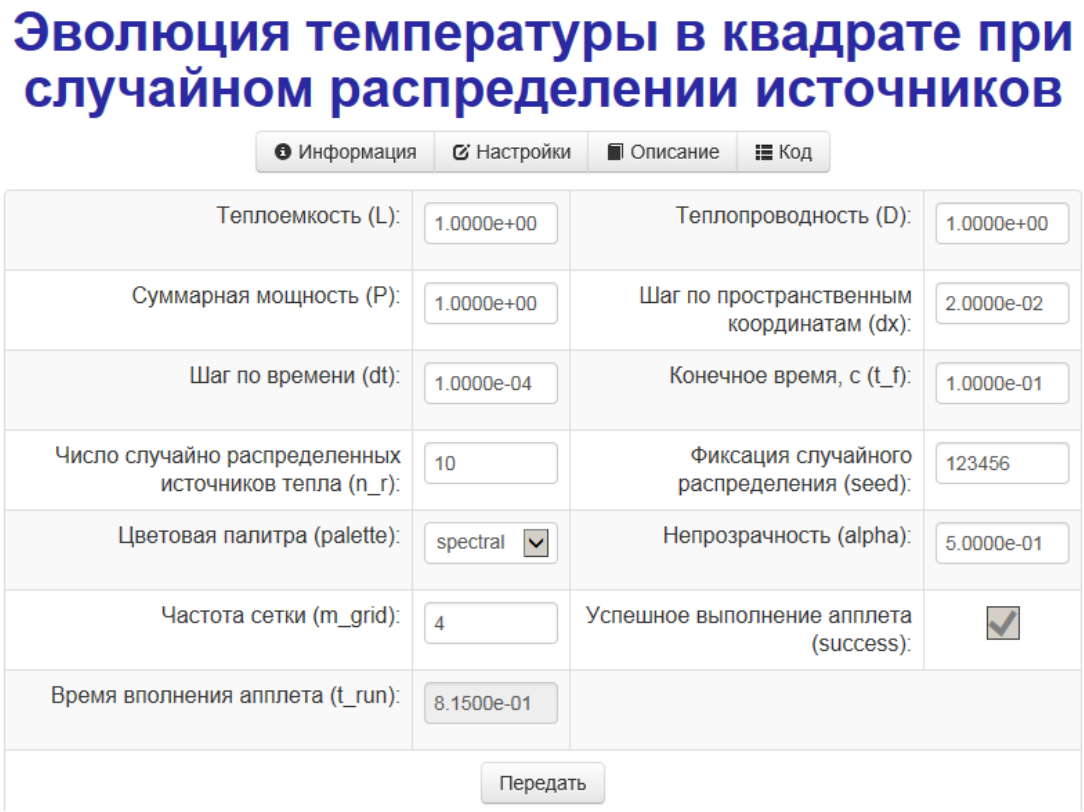

Рис. 10. — Пользовательский интерфейс апплета переменных Trm, автоматически сгенерированный по описаниям переменных

На четвертом шаге пользователь должен определить два метода: run() и check(). Кодирование метода run() является обязательным, в нем разработчик должен преобразовать значения входных переменных в значения выходных. На рис. 11 представлен фрагмент реализации метода run для апплета Trm.

```
def run(self, aux=dict()):
        t_start = time.time()
        # коэффициенты
        q1 = self.D*self.dt/self.L/self.dx**2
        q2 = 4.0*q1 -1.0
        q3 = self.dt/self.L

        n_x = int(np.round(1.0/self.dx+1, 0))
        n_t = int(np.round(self.t_f/self.dt+1, 0))

        # массивы
u0 = np.zeros((n_x, n_x))
        u1 = np.zeros((n_x, n_x))
        p  = np.zeros((n_x, n_x))

# источники тепла
        if self.seed >0:
            # фиксированное распределение
            rs = np.random.RandomState(self.seed)
px = rs.randint(1, n_x, self.n_r)
            py = rs.randint(1, n_x, self.n_r)
        else:
            px = np.random.randint(1, n_x, self.n_r)
            py = np.random.randint(1, n_x, self.n_r)
…
```

Рис. 11.— Фрагмент реализации метода run() апплета Trm

Из рис. 11 видно, что интерфейсные переменные доступны разработчику в виде атрибутов экземпляра класса, например, self.D, self.L.

Отображение результатов в графической форме осуществляется средствам библиотеки matplotlib. Перед ее использованием необходимо вызвать метод picstart(), завершается формирование картинки вызовом метода picfinish() без параметров.

По умолчанию осуществляется вывод картинки размером 400 на 400 пикселов с разрешением 96 пикселов на дюйм. Эти умолчания можно переопределить в методе picstart() с помощью именованных параметров width — ширина картинки в пикселах, height — высота картинки в пикселах, dpi — разрешение в пикселах на дюйм. На рис. 12 представлен фрагмент метода run() апплета Trm, отвечающей за вывод графики.

Для вывода кириллических надписей необходимо вызвать функцию cyrfont(), которая обеспечивает отображение кириллицы в надписях на графиках (строки 1–2). Объект картинки fig задается шириной 800 пикселов (строка 3). В данном случае на картинке отображаются два трехмерных графика (распределение источников тепла и распределение температуры), их горизонтальное расположение задается в строках 07–08.

```
01   from cyrfont import cyrfont
02   cyrfont()
03   fig = self.picstart(width=800)
04   x = np.linspace(0,1, n_x)
05   X,Y = np.meshgrid(x,x)
06
07   ax_p = fig.add_subplot(121, projection='3d')
08   ax_v = fig.add_subplot(122, projection='3d')
09
10   surf_p = ax_p.plot_surface(X,Y,p, rstride=self.m_grid,\
11   cstride=self.m_grid, alpha=self.alpha, cmap = \
12   plt.cm.get_cmap(self.palette))
13
14   surf_v = ax_v.plot_surface(X,Y,v1, rstride=self.m_grid, \
15       cstride=self.m_grid, alpha=self.alpha, cmap = \
16       plt.cm.get_cmap(self.palette))
17
18   # заголовки
19   ax_p.set_title(u'Распределениеисточников',\
20   fontweight='bold')
21   ax_v.set_title(u'Распределение температуры',\
22   fontweight='bold')
23
24   ax_p.set_xlabel('X')
25   ax_p.set_ylabel('Y')
26   ax_p.set_zlabel('P')
27
28   ax_v.set_xlabel('X')
29   ax_v.set_ylabel('Y')
30   ax_v.set_zlabel('u')
31
32   ticks = ax_p.get_xticklabels() + ax_p.get_yticklabels() + \
33   ax_p.get_zticklabels()+ax_v.get_xticklabels() + \
34   ax_v.get_yticklabels() + ax_v.get_zticklabels()
35
36   for item in ticks:
37       item.set_fontsize(8)
38
39   fig.tight_layout()
40   self.picfinish()
```

Рис.12.— Вывод графиков средствами библиотеки matplotlib

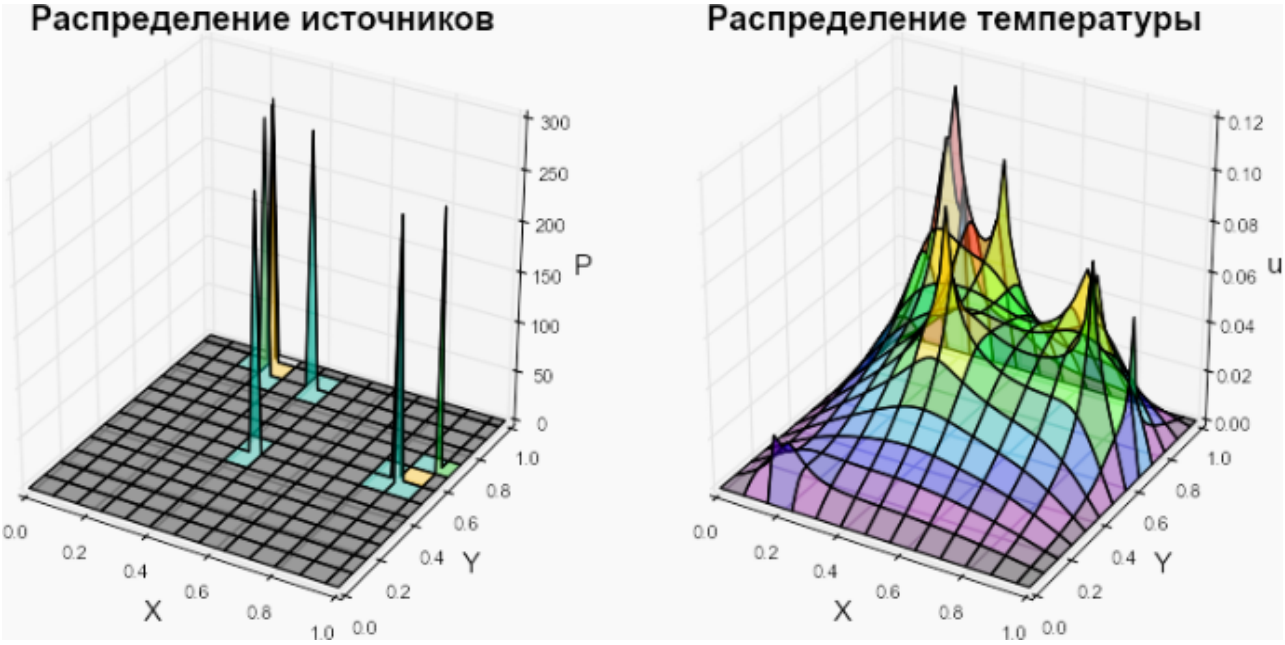

Рис. 13. — Графики в пользовательском интерфейсе апплета

Далее осуществляется вывод на графиках двух поверхностей (строки 10–12 и 13–16), «раскрашивание» поверхностей осуществляется с помощью палитр, выбранных пользователем. Вывод заголовков графиков, надписей на осях осуществляется в строках 19–34. При отладке оказалось, что оцифровка осей слишком крупная, поэтому размер шрифтов для оцифровки был изменен в строках (32–37), кроме того, вместо стандартной раскладки графиков был выбрана «сжатая» раскладка (строка 39). Результаты в пользовательском интерфейсе апплета представлены на рис. 13.

Следует отметить, что апплеты отличаются от стандартных приложений Python только необходимостью определения интерфейсных переменных в виде атрибутов класса, а также вызовом методов picstart() и picfinish() для вывода графиков.

Если необходимо осуществлять нетривиальные проверки сочетаний значений переменных, необходимо реализовать метод check(), который в случае удачного завершения проверок возвращает пустую строка, а в случае неудачного — сообщение об ошибке. Если проверки не проводятся, то метод check() можно не реализовывать, как это сделано в апплете Trm.

Разработка и отладка апплетов осуществляется в интегрированной среде PyScripter, входящей в комплект поставки Portable Python. Однако можно использовать и другие интегрированные среды разработки, включая Wings, PyCharm и даже Visual Studio в сочетании с Python Tools for Visual Studio 2.0. Для разработки можно использовать специальный отладочный стенд stand.py, который в локально воспроизводит среду выполнения апплетов.

Одновременно с разработкой самого апплета рекомендуется публиковать его описание и исходный код. Для этого достаточно скомпоновать описание в Microsoft Word, сохранить его в формате html в каталоге applets/static/texts/,

скопировать в этот же каталог исходный текст апплета, задать ссылки в описании апплета, как это показано на рис. 7.

ЗАКЛЮЧЕНИЕ

Резюмируя сказанное выше, можно отметить, что апплеты представляют собой удобную среду для разработки и применения в учебном процессе небольших расчетных веб-приложений. Процесс разработки апплетов практически не отличается от создания консольных приложений Python, позволяет использовать пакеты и модули библиотек Python, включая numpy, scipy, matplotlib, alglib.

Трудоемкость создания апплетов сопоставима с трудоемкостью создания приложений на основе Matlab, Mapple, Mathematica, но по сравнению с ними можно работать с приложениями не только в локальной среде, но и публиковать их в веб. Среда разработки и выполнения апплетов является полностью переносимой, не требует установки на рабочем месте разработчика и пользователя. Использование языка программирования Python не ограничивает возможности системы встроенными в нее библиотеками и допускает возможность расширения за счет имеющихся в открытом доступе пакетов и библиотек.

Разработка апплетов не требует высокой квалификации, может осуществляться как преподавателями, так и студентами. Полный пример всех этапов разработки апплета представлен в виде видео, входящего в комплект поставки веб-приложения апплетов.

Одновременно с апплетами можно осуществлять интегрированную публикацию электронных образовательных ресурсов, включая html-, pdf-файлы, видео, исходные тексты программ на различных языках. Авторы используют средства для разработки апплетов при подготовке к практическим занятиям и при разработке электронных учебно-методических комплексов по своим курсам.

ЛИТЕРАТУРА

1. Данилов О. Л., Гаряев А. Б., Яковлев И. В., Клименко А. В., Вакулко А. Г. Энергосбережение в теплоэнергетике и теплотехнологиях/ Данилов О. Л., Гаряев А. Б., Яковлев И. В., Клименко А. В., Вакулко А. Г.; под ред. Клименко А.В. – М.: Издательский дом МЭИ, 2011. – 424.
2. Прохоренок Н.А. Python.Самое необходимое. – Санкт-Петербург: БХВ-Петербург, 2011, – 416 с.
3. IdrisI. Numpy Beginner’s Guide. Second Edition. – Birmingham: PACT Publishing, 2013. – 310 p.
4. Tosi S. Matplotlib for Python Developers. – Birmingham: PACT Publishing, 2009. – 307 p.
5. DiPierroM. WEB2PY. CompleteReferenceforSolutions.– Chicago: EXPERTS4SOLUTIONS, – 2013, 614. – Режимдоступа: https://dl.dropboxusercontent.com/u/18065445/web2py/web2py_manual_5th.pdf. Дата доступа: 12.10.13