Апплеты

Лекция 8

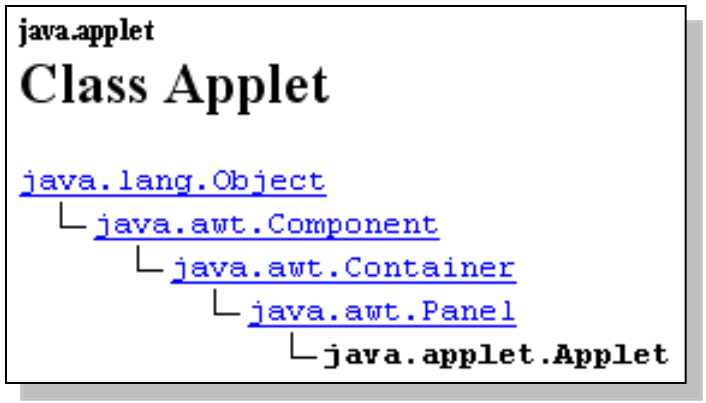
Цели

- Дать определение апплета
- Объяснить различия между приложениями апплетами Java
- Создать апплет
- Определить, каким образом параметры передаются в апплеты
- Описать обработку событий с помощью апплетов

Апплеты

- **Апплет (`Applet`)** – это Java-программа, которая может быть встроена в HTML-страницу и выполняется в Java-совместимом браузере.
- Создаётся наследованием от класса **java.applet.Applet**
- Примеры Java-совместимых web-браузеров - ?

Апплеты не запускаются JVM — их загружает браузер, который сам запускает JVM для выполнения апплета. Эти особенности отражаются на написании программы апплета.



Безопасность апплета

- Апплеты были разработаны, чтобы загружаться на отделенной стороне и затем выполняться локально.
- Безопасность – жизненная проблема для апплетов.
- Если пользователь позволяет коду работать, браузер загрузит весь код апплета и выполнит его немедленно.
- Пользователь не имеет возможности подтвердить или остановить выполнение апплета.
- По этой причине апплеты, в отличие от приложений, ограничены в действиях.
- Апплет, загруженный в сети, работает в системе пользователя в безопасной среде, названной **средой исполнения**.

Безопасность апплета

- Апплет
  - не может обращаться к файловой системе машины, на которой он выполняется, даже для чтения файлов или просмотра каталогов;
  - может связаться по сети только с тем сайтом, с которого он был загружен;
  - не может прочитать системные свойства;
  - не может печатать на принтере, подключенном к тому компьютеру, на котором он выполняется;
  - не может воспользоваться буфером обмена (clipboard);
  - не может запустить приложение методом exit();
  - не может остановить JVM методом exit();
  - не может создавать классы в пакетах java.*, а классы пакетов sun.* не может даже загружать.

Безопасность апплета

- Кроме того, Java всеми возможными способами защищает систему пользователя от повреждения или заражения вирусом.
- В первую очередь, осуществляется проверка целостности байт-кода, поступающего на компьютер. Если в процессе передачи код был изменен, виртуальная Java-машина не будет его запускать.
- Браузеры могут усилить или ослабить свои ограничения, например, разрешить локальным апплетам, загруженным с той же машины, где они выполняются, доступ к файловой системе.
- Наименьшие ограничения имеют **доверенные** (**trusted**) **апплеты**, снабженные электронной подписью с помощью классов из пакетов **java.security.*.**

Различия между апплетами и приложениями

- **Апплет в основном предназначается для развёртывания в web.**
  Приложения предназначаются для работы в качестве автономной программы.

- **Апплеты создаются посредством расширения класса java.applet.Applet или javax.swing.JApplet**
  Для приложений не существует никаких ограничений.

- **Апплеты работают в любом совместимом браузере.**
  Приложения работают с помощью интерпретатора Java.

Различия между апплетами и приложениями

- **Выполнение апплетов начинается с метода `init()`.**

  Выполнение приложений начинается с метода `main()`.

- **Апплет обязательно должен содержать по меньшей мере один *public* класс, в противном случае компилятор сообщит об ошибке. В апплете не обязательно определять метод main().**

  В приложении метод main() обязательно должен быть включён в public класс.

Жизненный цикл апплета

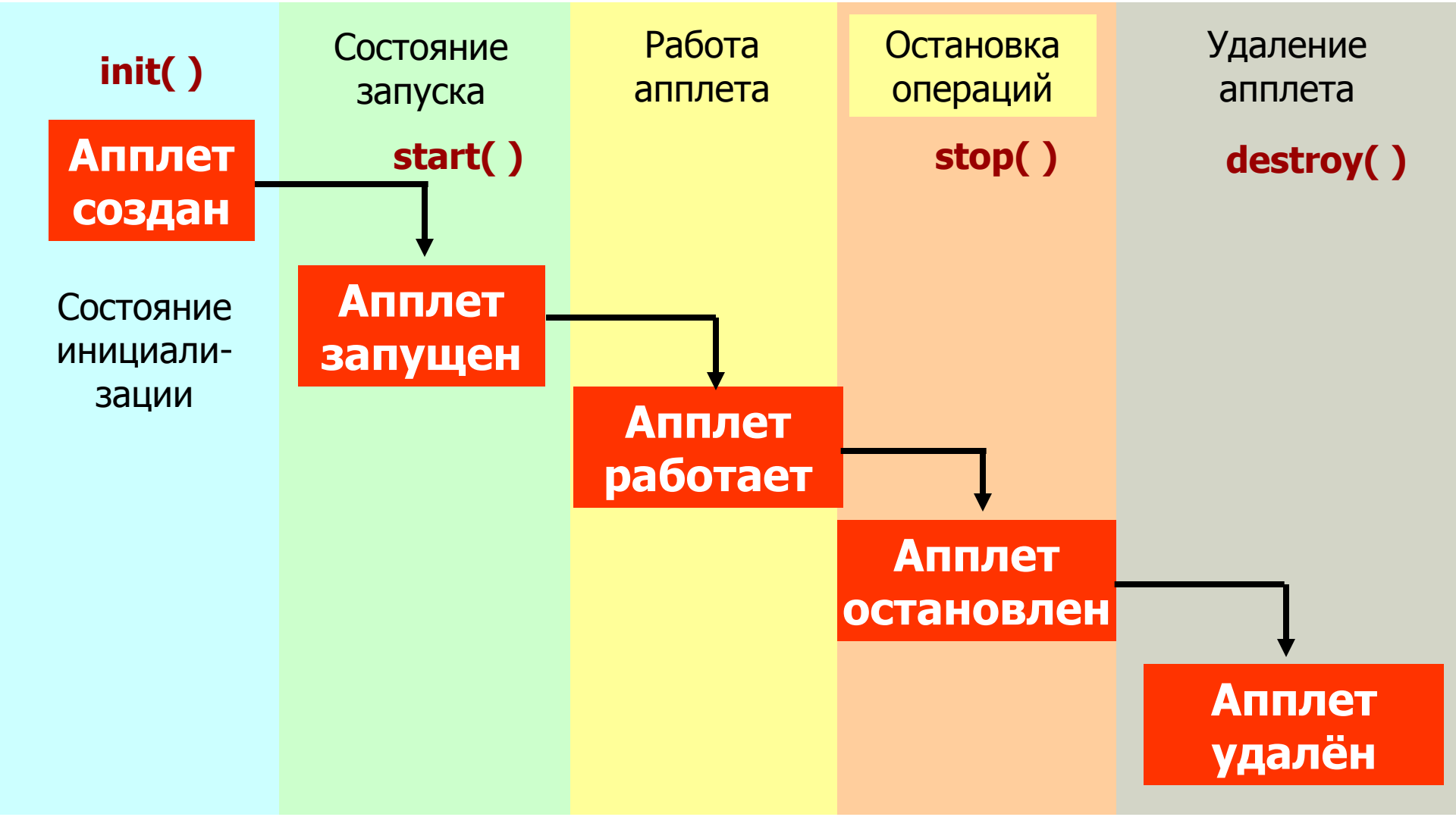
- **Жизненный цикл** объекта определяет этапы, которые он должен пройти, начиная с его создания и до его уничтожения.
- Апплет определяет свою структуру, исходя из четырёх событий, которые происходят во время выполнения.
- Для каждого события автоматически вызывается соответствующий метод.

Жизненный цикл апплета

- Методы перечислены ниже:

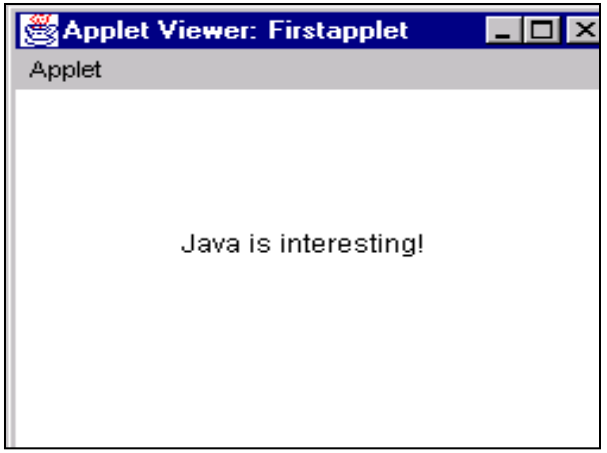
Метод	Операция
init()	Автоматически вызывается для выполнения начальной инициализации апплета, включая компоновку компонент. Вы всегда перегружаете этот метод.
start()	Вызывается каждый раз, когда апплет переносится в поле зрения Web-браузера, чтобы позволить апплету начать нормальные операции (особенно те, которые останавливаются в методе stop()). Также вызывается после init().
stop()	Вызывается каждый раз, когда апплет выходит из поля зрения Web-браузера, чтобы позволить апплету завершить дорогостоящие операции. Также вызывается перед destroy().
destroy()	Вызывается тогда, когда апплет начинает выгружаться со страницы для выполнения финального освобождения ресурсов, когда апплет более не используется.

Жизненный цикл апплета



Простой апплет

```
import java.awt.*;
import java.applet.*;
public class FirstApplet extends Applet
{
    String str;
    public void init() {
        str = "Java is interesting!";
    }
    public void paint(Graphics g)
    {
        g.drawString(str, 70, 80);
    }
}
```



Создание апплета

- Апплет компилируется с помощью компилятора Java : `javac`
  - `javac Firstapplet.java`

- Создание HTML-страницы для вывода апплета

```
<html>
    <applet
        code=Firstapplet.class
        width=200
        height=200 >
    </applet>
</html>
```

- В состав JDK любой версии входит программа **appietviewer**:
  - `appletviewer Firstapplet.html`

Создание апплета

Обязательные параметры тега <applet>**:**

- **code** — URL-адрес файла с классом апплета или архивного файла;
- **width** и **height** — ширина и высота апплета в пикселах.

Необязательные параметры:

- **codebase** — URL-адрес каталога, в котором расположен файл класса апплета. Если этот параметр отсутствует, браузер будет искать файл в том же каталоге, где размещен соответствующий HTML-файл;
- **archive** — файлы всех классов, составляющих апплет, могут быть упакованы архиватором ZIP или специальным архиватором JAR в один или несколько архивных файлов. Параметр задает URL-адреса этих файлов через запятую;
- **align** — выравнивание апплета в окне браузера. Этот параметр имеет одно из следующих значений: ABSBOTTOM, ABSMIDDLE, BASELINE, BOTTOM, CENTER, LEFT, MIDDLE, RIGHT, TEXTTOP, TOP;
- **hspace** и **vspace** — горизонтальные и вертикальные поля, отделяющие апплет от других объектов в окне браузера в пикселах;
- **style** — информация о стиле CSS (Cascading Style Sheet);
- **alt** — текст, выводимый вместо апплета, если браузер не может загрузить его.
- …

Вывод изображений с помощью апплетов

```
/*
<applet code = DisplayImage width = 200 height = 200>
</applet>
*/
import java.awt.*;
import java.applet.*;
public class DisplayImage extends Applet
{
    Image img;
    public void init() {
        img = getImage(getCodeBase(),"duke.gif");
    }
    public void paint(Graphics g) {
        g.drawImage(img,20,20,this);
    }
}
```

Вывод изображений с помощью апплетов

- Для вывода изображений необходимо воспользоваться классами **Image** и **Graphics**.
- Метод **`getCodeBase()`** получает базовый URL апплета
- Метод **`getImage()`** возвращает объект `Image`, который может быть прорисован на экране.
- Метод **`drawImage()`** принимает четыре параметра – объект `Image`, положение в виде координат x и y, а также объект типа `ImageObserver`.

Передача параметров

- Параметры позволяют пользователю управлять определёнными свойствами апплета.
- Для передачи управляющей информации в апплет используются параметры.
- Например, можно передать имя и размер изображения для динамического обновления, цвет тип и размер шрифта и т.д.
- Передача параметров производится с помощью тегов <param>, располагаемых между тегами <applet> и </applet> в HTML-файле.

Передача параметров

- В тегах <param> указывается название параметра name (регистр в имени не учитывается) и его значение value.

```
<param name = "fontName" value = "Serif">
<param name = "fontSize" value = "10">
```

- Апплет получает параметры с помощью метода **getParameter** класса Applet, обычно ее вызов размещают в функции инициализации:

public String getParameter(String name) - возвращает значение параметра value в виде строки аргумента задается параметра name, причем регистр букв не различается.

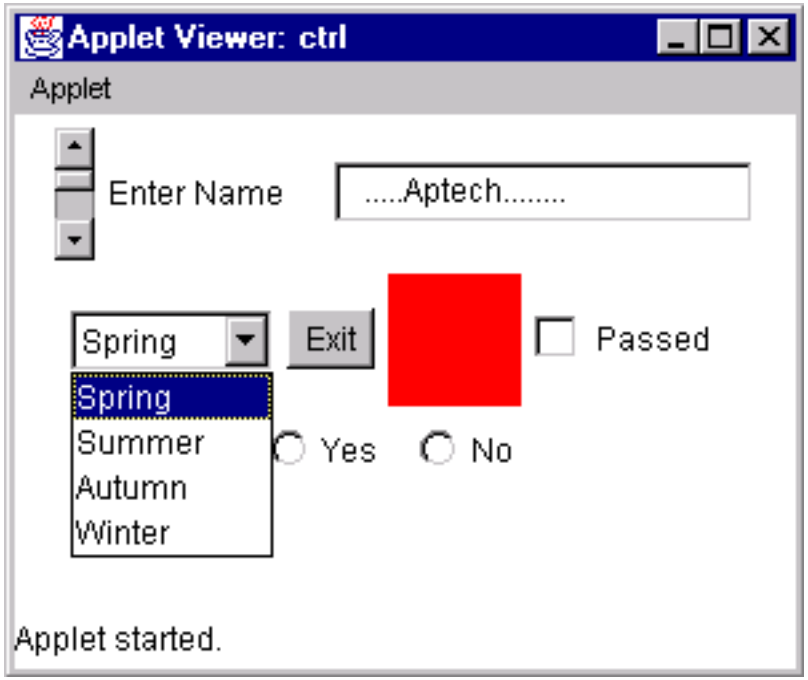
Пример

```
<html>
<applet code = ImageDemo.class width = 150  height = 150>
   <param name = "image" value = "duke.gif">
</applet>
</html>
```

```
import java.awt.*;
import java.applet.*;
public class ImageDemo extends Applet {
   Image img;
   public void init() {
      String imagename = getParameter("image");
      img = getImage(getCodeBase(),imagename);
   }
   public void paint(Graphics g) {
      g.drawImage(img,20,20,this);
   }
}
```

Апплеты и GUI

- **Графический интерфейс пользователя** (GUI) используется для создания наглядного визуального интерфейса, с которым легко работать.
- По умолчанию макетом апплета является **FlowLayout**.
- Различные элементы управления, которые могут быть созданы:

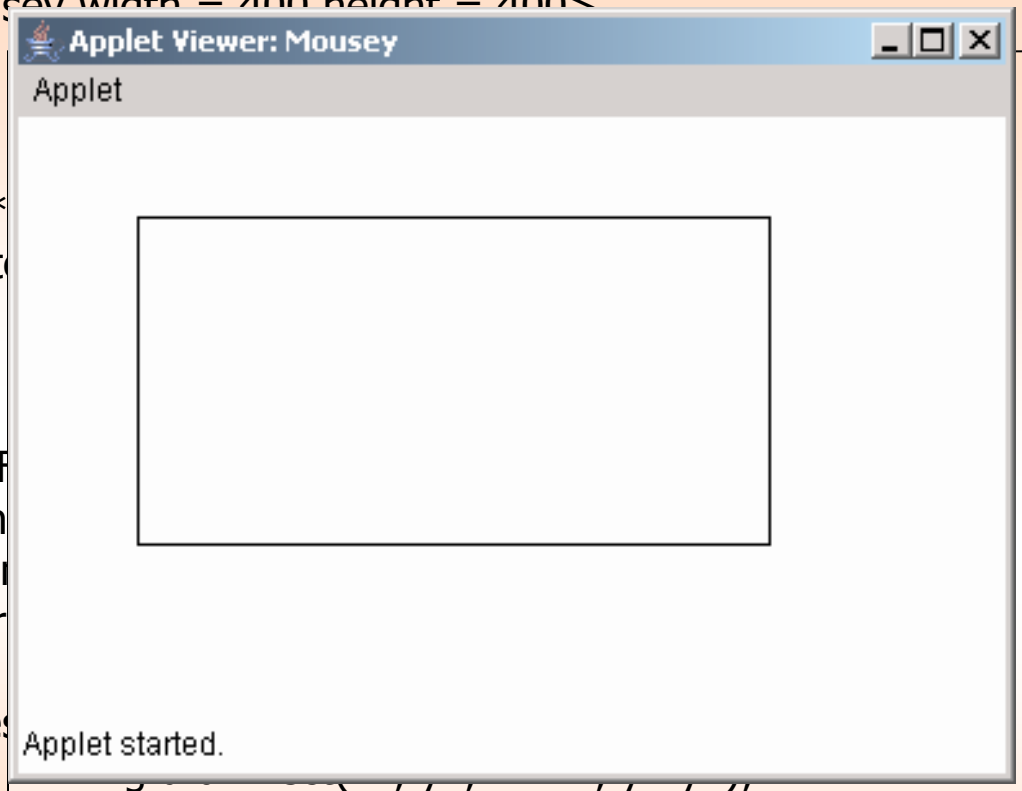


Обработка событий

- Для работы с событиями требуются обработчики, которые должны реагировать на события надлежащим образом.
- Процедура, которую необходимо выполнять при генерации события:
  - определить тип события;
  - определить компонент, который сгенерировал событие;
  - написать требуемый код для обработки события.

Пример

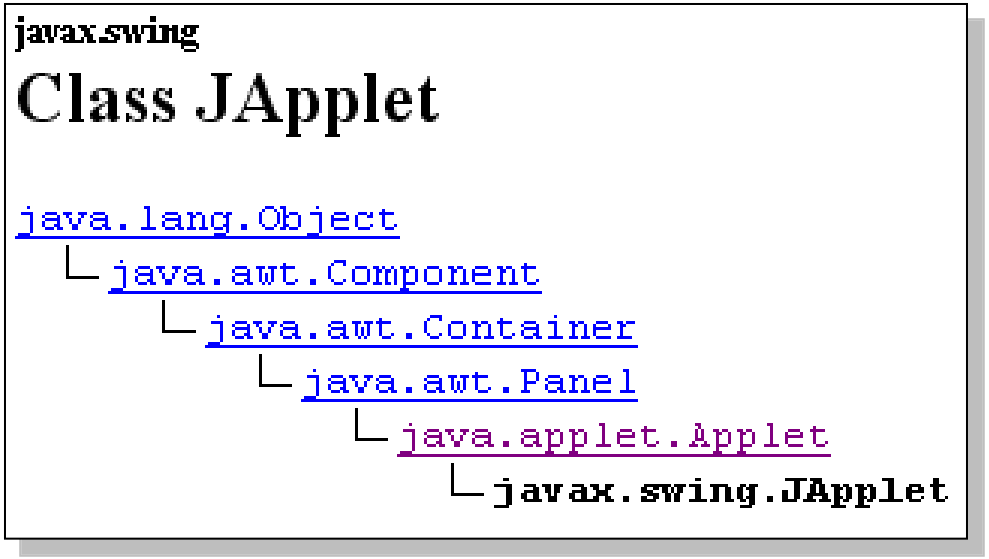
```
/* <applet code = Mousey width = 400 height = 400>
</applet>*/
import java.awt.*;
import java.applet.*;
import java.awt.event.*
public class Mousey ext
    int x1, y1, x2, y2;
    public void init()
    {
            setLayout(new F
            addMouseListen
            addMouseMotion
            this.setVisible(tr
    }
    public void mousePres
        x1 = e.getX();
        y1 = e.getY();
    }
```




```
        x2 = 0;
        y2 = 0;
    }
}
```

Класс JApplet

- Производным класса Applet из AWT является Swing-класс JApplet.



- Соответственно, данный класс также позволяет создавать апплеты, однако имеет ряд преимуществ.