

Java Overview

An introduction to the Java Programming Language

Produced Eamonn de Leastar (edeleestar@wit.ie)
by: Dr. Siobhan Drohan (sdrohan@wit.ie)

Essential Java

⊕ Overview

- ⊕ Introduction
- ⊕ Syntax
- ⊕ Basics
- ⊕ Arrays

⊕ Classes

- ⊕ Classes Structure
- ⊕ Static Members
- ⊕ Commonly used Classes

⊕ Control Statements

- ⊕ Control Statement Types
- ⊕ If, else, switch
- ⊕ For, while, do-while

⊕ Inheritance

- ⊕ Class hierarchies
- ⊕ Method lookup in Java
- ⊕ Use of this and super
- ⊕ Constructors and inheritance
- ⊕ Abstract classes and methods

Interfaces

⊕ Collections

- ⊕ ArrayList
- ⊕ HashMap
- ⊕ Iterator
- ⊕ Vector
- ⊕ Enumeration
- ⊕ Hashtable

⊕ Exceptions

- ⊕ Exception types
 - ⊕ Exception Hierarchy
 - ⊕ Catching exceptions
 - ⊕ Throwing exceptions
 - ⊕ Defining exceptions
- Common exceptions and errors

⊕ Streams

- ⊕ Stream types
- ⊕ Character streams
- ⊕ Byte streams
- ⊕ Filter streams
- ⊕ Object Serialization

Overview: Road Map

⊕ Java Introduction

⊕ History

⊕ Portability

⊕ Compiler

⊕ Java Virtual
Machine

⊕ Garbage collection

⊕ Java Syntax

⊕ Identifiers

⊕ Expressions

⊕ Comments

⊕ Java Basics

⊕ Java types

⊕ Primitives

⊕ Objects

⊕ Variables

⊕ Operators

⊕ Identity and
equality

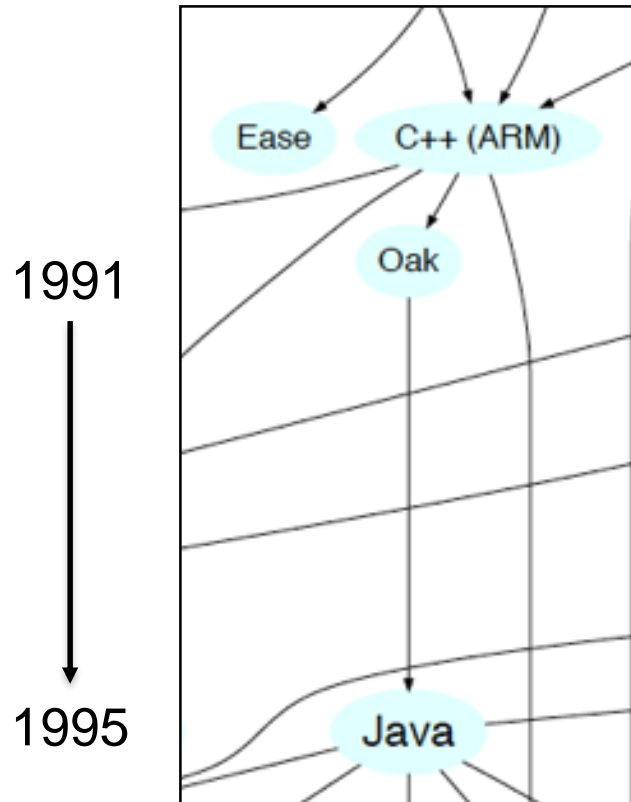
⊕ Arrays

⊕ What are arrays?

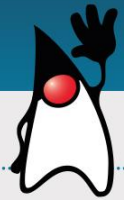
⊕ Creating arrays

⊕ Using arrays

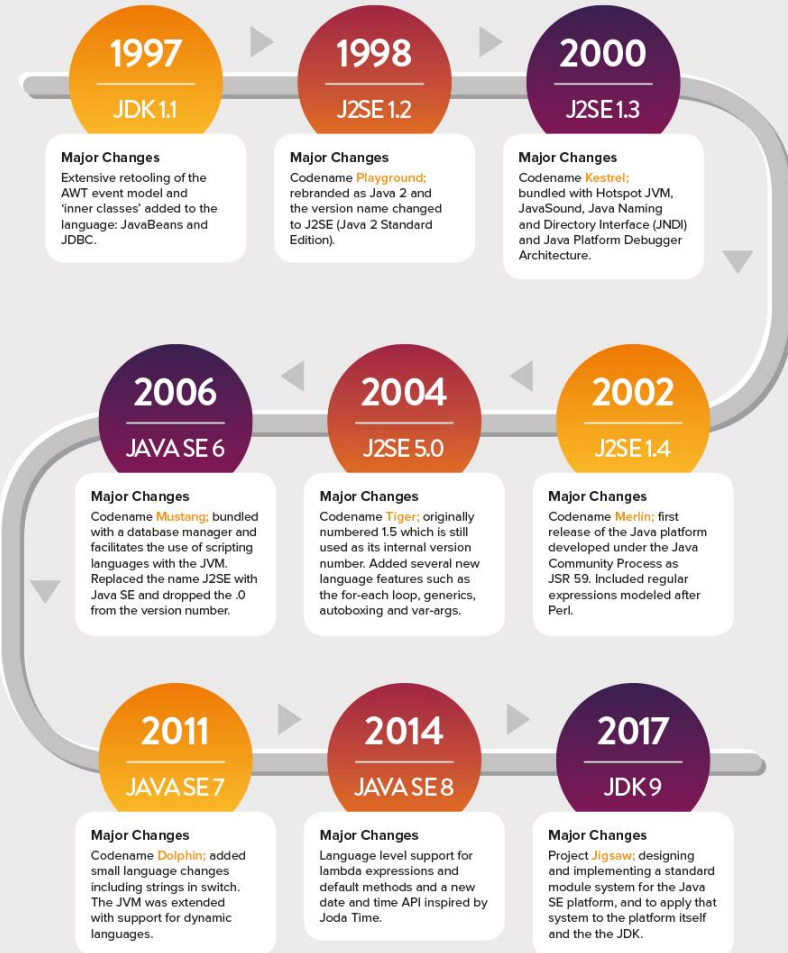
Java History



A SHORT HISTORY OF JAVA



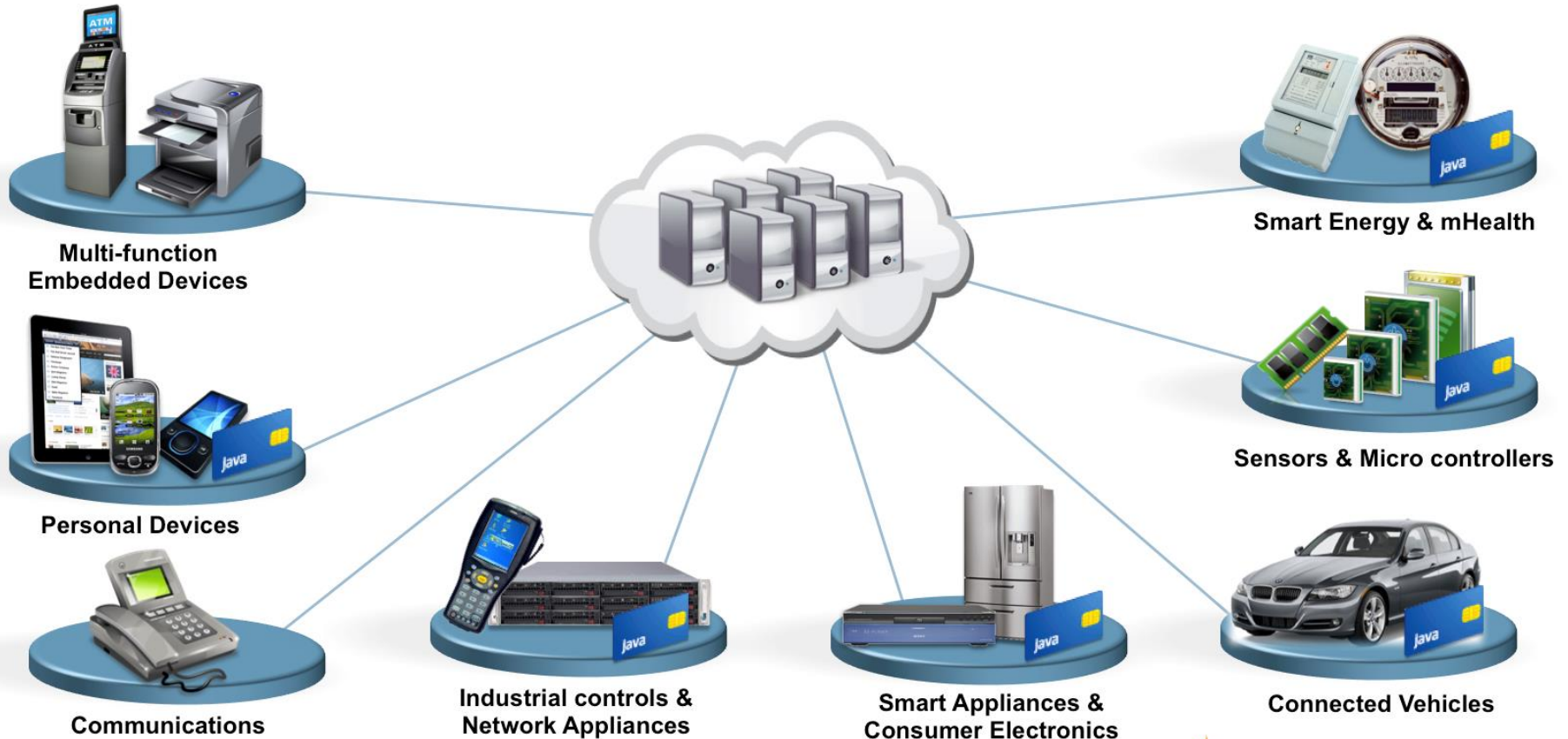
JAVA VERSION HISTORY



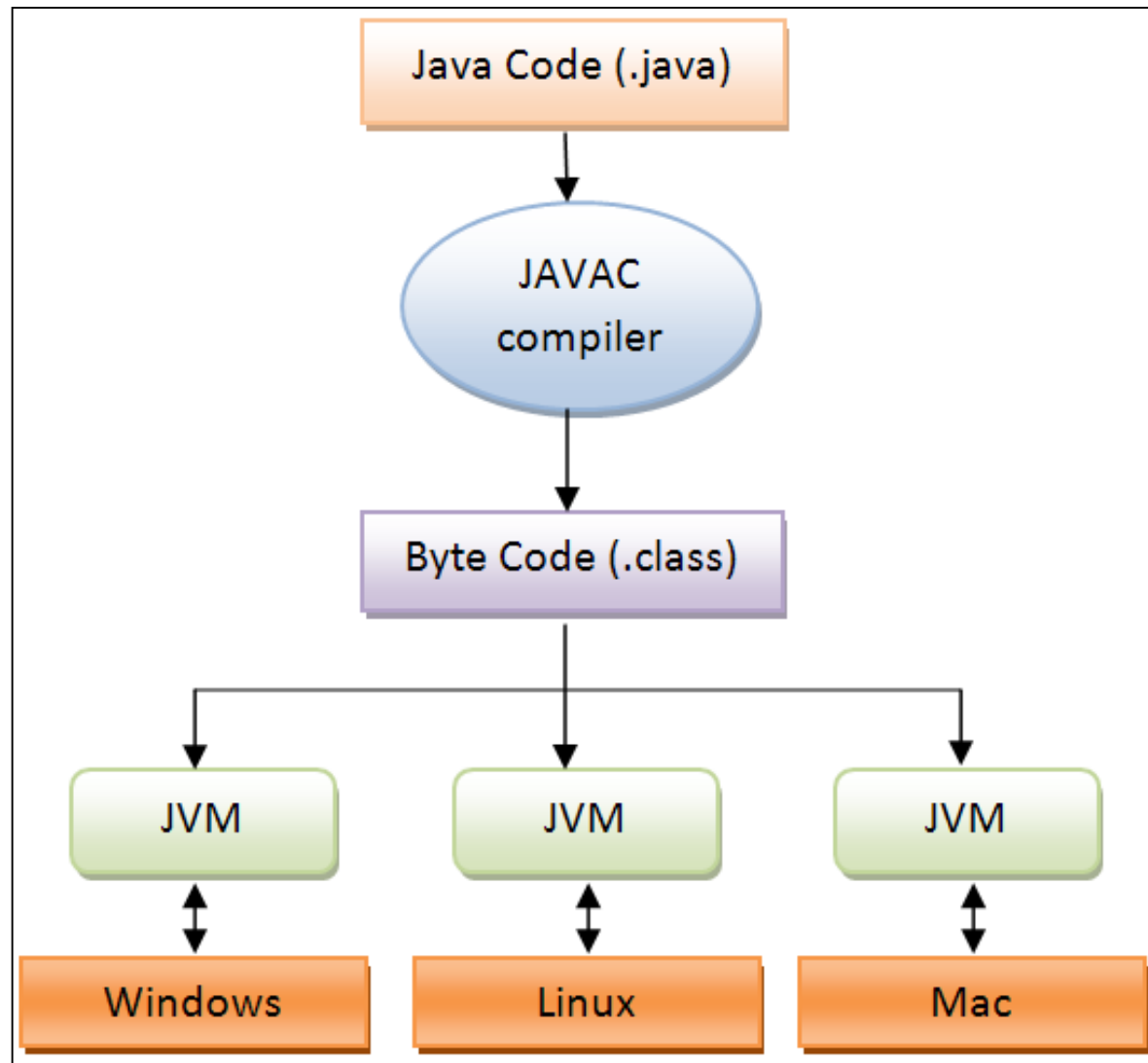
Initially intended for:



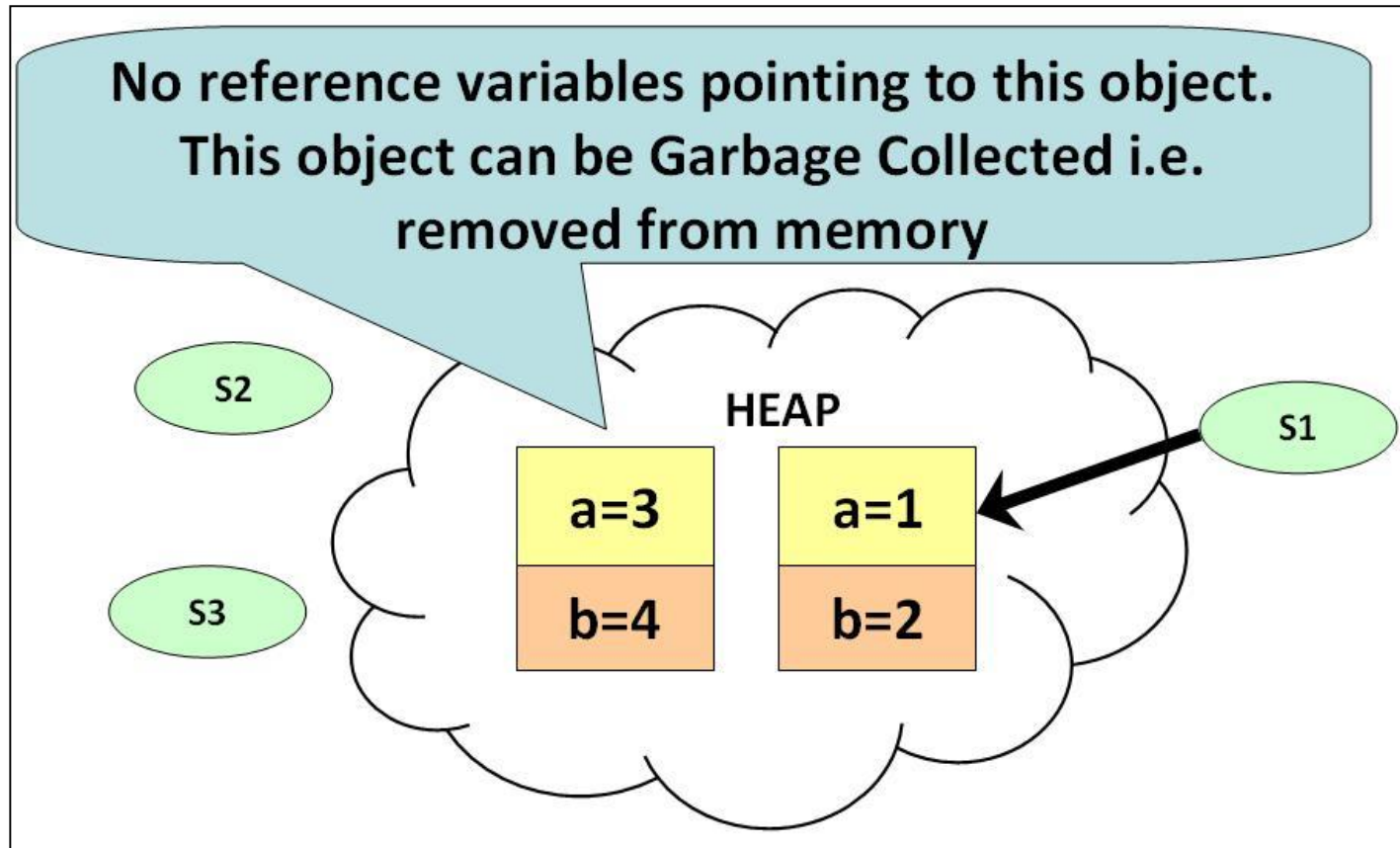
Java Embedded



Portability / Compiler / JVM

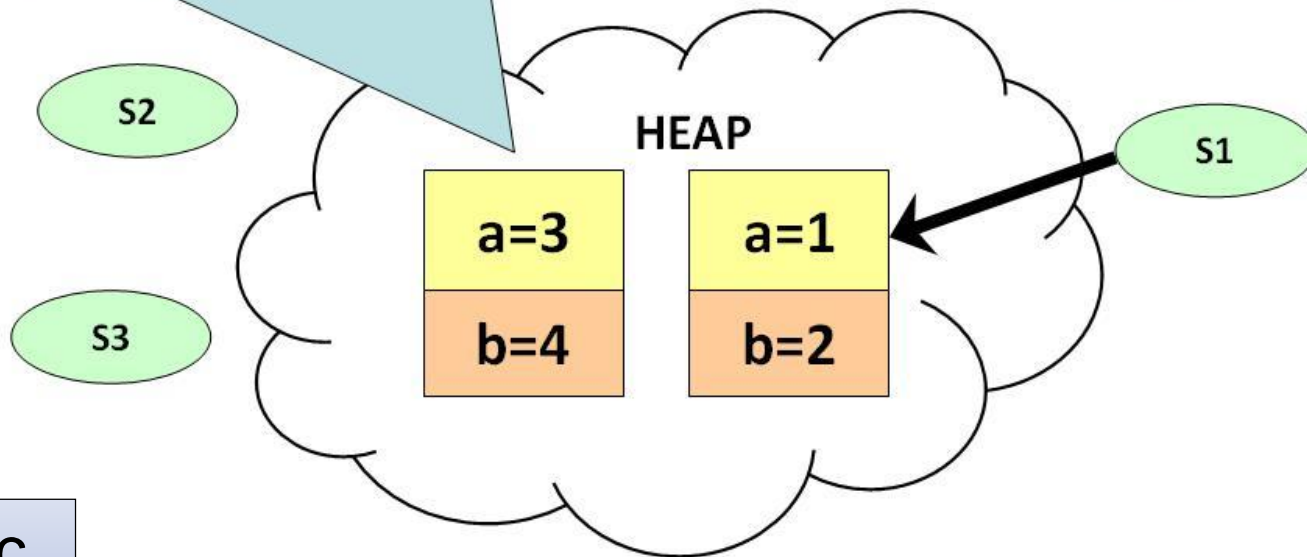


Memory Management



Memory Management

**No reference variables pointing to this object.
This object can be Garbage Collected i.e.
removed from memory**



Automatic

Happens when memory is required

Can be forced programmatically

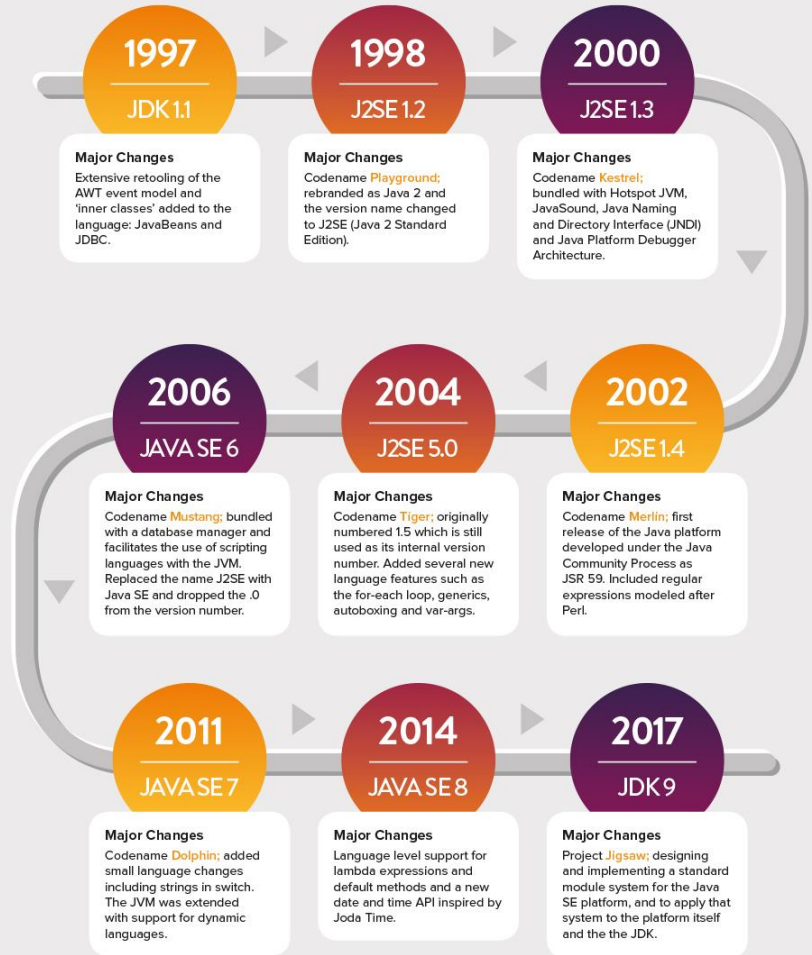
Java Cadence

- Initially had releases every two/three years.
- Now the cadence is every 6 months:
 - JDK9 → Sept 2017
 - JDK10 → March 2018
 - JDK11 → Sept 2018

A SHORT HISTORY OF JAVA



JAVA VERSION HISTORY



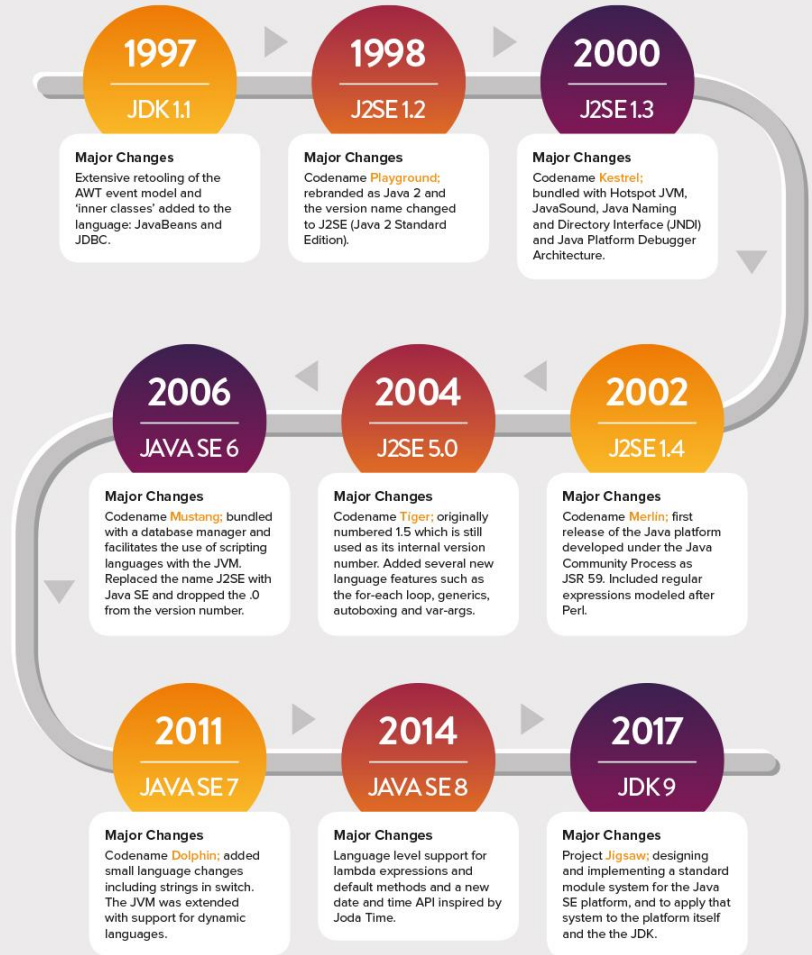
Java Cadence

Week(s)	Version
1 – 5	Focus on Java 7 constructs
6 +	Explore some of Java 8, 9, 10 and 11 changes. This will give us a good window into the evolution of the Java language.

A SHORT HISTORY OF JAVA



JAVA VERSION HISTORY



Overview: Road Map

⊕ Java Introduction

- ⊕ History

- ⊕ Portability

- ⊕ Compiler

- ⊕ Java Virtual
Machine

- ⊕ Garbage collection

⊕ Java Syntax

- ⊕ Identifiers

- ⊕ Expressions

- ⊕ Comments

⊕ Java Basics

- ⊕ Java types

- ⊕ Primitives

- ⊕ Objects

- ⊕ Variables

- ⊕ Operators

- ⊕ Identity and
equality

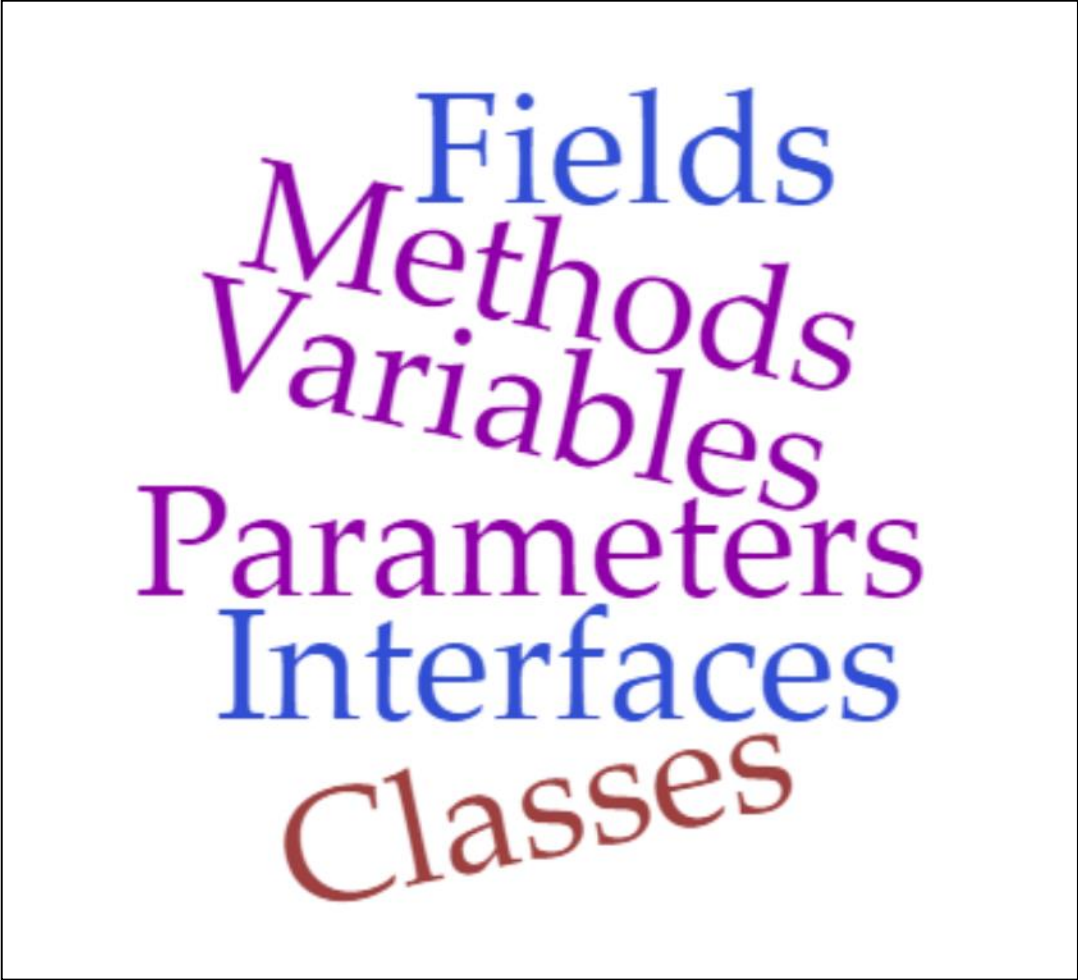
⊕ Arrays

- ⊕ What are arrays?

- ⊕ Creating arrays

- ⊕ Using arrays

Identifiers are used for naming:



Fields
Methods
Variables
Parameters
Interfaces
Classes

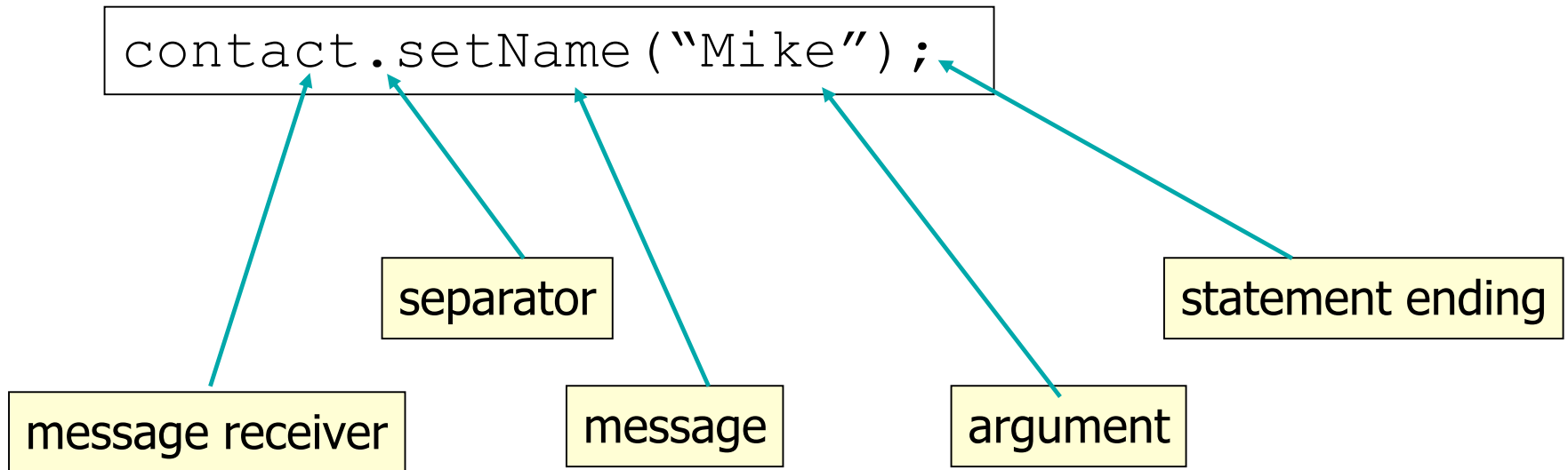
Identifiers

- Are case-sensitive.
- Begin with either:
 - a **letter (preferable)**,
 - the dollar sign "\$", or
 - the underscore character "_".
- Can contain letters, digits, dollar signs, or underscore characters.
- Can be any length you choose.
- Must not be a **keyword or reserved word** e.g. int, while, etc.
- Cannot contain white spaces.

Identifiers

Variable Name	Remarks
speed	Valid variable name
_speed	Valid but bad variable name
\$speed	Valid but bad variable name
speed1	Valid variable name
spe ed	Invalid variable name
spe"ed	Invalid variable name

Messages and Objects



Statements → Basic Java Expressions

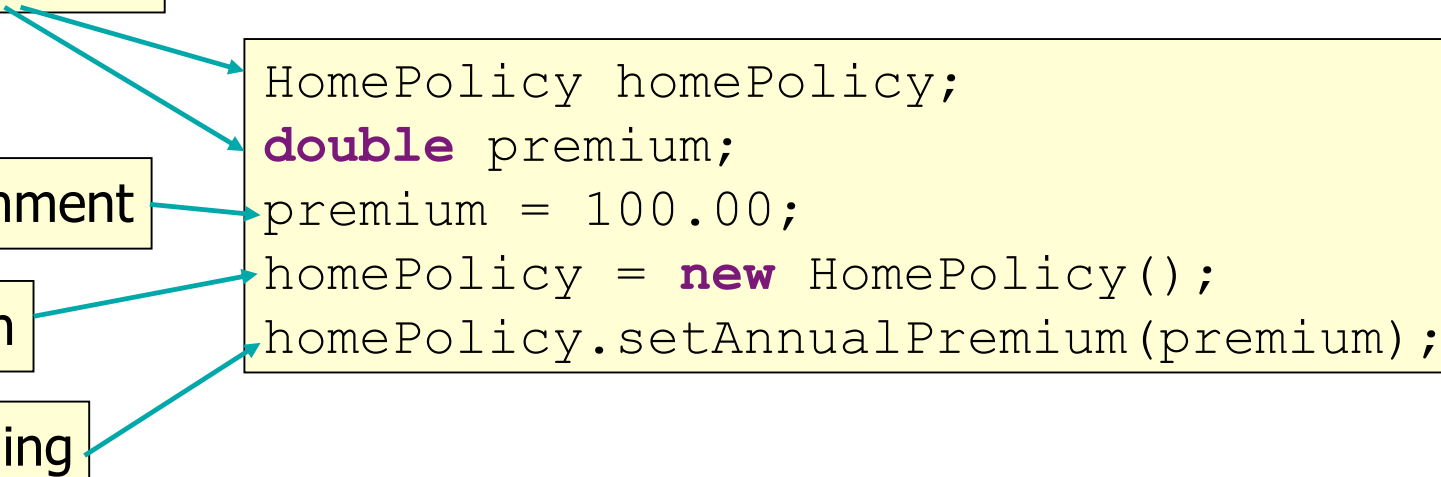
variable declaration

variable assignment

object creation

message sending

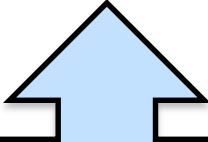
```
HomePolicy homePolicy;  
double premium;  
premium = 100.00;  
homePolicy = new HomePolicy();  
homePolicy.setAnnualPremium(premium);
```



Empty Expression

```
;    //this is an empty statement  
    // on its own in the line  
    //it means...do nothing!
```

```
for(int i=1; i<3; i++) ;  
    System.out.println(i);
```



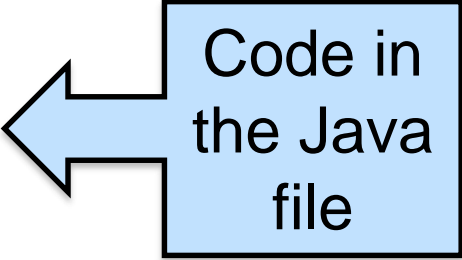
We would expect 1, 2 printed but it only prints 1 because of the statement terminator.

Comments

```
/** Javadoc example comment.  
 * Used for generation of the documentation.  
 */  
  
/* Multiple line comment.  
 *  
 */  
  
// Single line comment.
```

Javadoc

Code in
the Java
file



```
/**
 * Returns an Image object that can then be painted on the screen.
 * The url argument must specify an absolute {@link URL}. The name
 * argument is a specifier that is relative to the url argument.
 * <p>
 * This method always returns immediately, whether or not the
 * image exists. When this applet attempts to draw the image on
 * the screen, the data will be loaded. The graphics primitives
 * that draw the image will incrementally paint on the screen.
 *
 * @param url an absolute URL giving the base location of the image
 * @param name the location of the image, relative to the url argument
 * @return the image at the specified URL
 * @see Image
 */
public Image getImage(URL url, String name) {
    try {
        return getImage(new URL(url, name));
    } catch (MalformedURLException e) {
        return null;
    }
}
```

Produces
this
HTML
code



getImage

```
public Image getImage(URL url,
    String name)
```

Returns an `Image` object that can then be painted on the screen. The `url` argument must specify an absolute URL. The `name` argument is a specifier that is relative to the `url` argument.

This method always returns immediately, whether or not the image exists. When this applet attempts to draw the image on the screen, the data will be loaded. The graphics primitives that draw the image will incrementally paint on the screen.

Parameters:

`url` - an absolute URL giving the base location of the image.

`name` - the location of the image, relative to the `url` argument.

Returns:

the image at the specified URL.

See Also:

`Image`

Java API → Javadoc output

The screenshot shows the Oracle Java Platform, Standard Edition 7 API Specification page. The browser address bar shows the URL <https://docs.oracle.com/javase/7/docs/api/>. The page has a navigation bar with tabs for Overview, Package, Class, Use, Tree, Deprecated, Index, and Help. The Overview tab is selected. On the left, there is a sidebar with 'All Classes' and 'Packages' sections. The 'Packages' section lists various Java packages like java.applet, java.awt, java.awt.color, etc. The main content area displays the title 'Java™ Platform, Standard Edition 7 API Specification' and a description: 'This document is the API specification for the Java™ Platform, Standard Edition.' Below this, there is a 'See: Description' link. A 'Packages' table is shown, listing various Java packages and their descriptions.

Package	Description
java.applet	Provides the classes necessary to create an applet and the classes an applet uses to communicate with its applet context.
java.awt	Contains all of the classes for creating user interfaces and for painting graphics and images.
java.awt.color	Provides classes for color spaces.
java.awt.datatransfer	Provides interfaces and classes for transferring data between and within applications.
java.awt.dnd	Drag and Drop is a direct manipulation gesture found in many Graphical User Interface systems that provides a mechanism to transfer information between two entities logically associated with presentation elements in the GUI.
java.awt.event	Provides interfaces and classes for dealing with different types of events fired by AWT components.
java.awt.font	Provides classes and interface relating to fonts.
java.awt.geom	Provides the Java 2D classes for defining and performing operations on objects related to two-dimensional geometry.
java.awt.im	Provides classes and interfaces for the input method framework.
java.awt.im.spi	Provides interfaces that enable the development of input methods that can be used with any Java runtime environment.
java.awt.image	Provides classes for creating and modifying images.
java.awt.image.renderable	Provides classes and interfaces for producing rendering-independent images.
java.awt.print	Provides classes and interfaces for a general printing API.
java.beans	Contains classes related to developing <i>beans</i> -- components based on the JavaBeans™ architecture.
java.beans.beancontext	Provides classes and interfaces relating to bean context.
java.io	Provides for system input and output through data streams, serialization and the file system.
java.lang	Provides classes that are fundamental to the design of the Java programming language.

Literals

```
String one = "One";  
String two = "Two";
```

=

```
String one = new String("One");  
String two = new String("Two");
```


What we covered in this lecture:

⊕ **Overview**

- ⊕ Introduction
- ⊕ Syntax

- ⊕ Basics
- ⊕ Arrays

⊕ **Classes**

- ⊕ Classes Structure
- ⊕ Static Members
- ⊕ Commonly used Classes

⊕ **Control Statements**

- ⊕ Control Statement Types
- ⊕ If, else, switch
- ⊕ For, while, do-while

⊕ **Inheritance**

- ⊕ Class hierarchies
- ⊕ Method lookup in Java
- ⊕ Use of this and super
- ⊕ Constructors and inheritance
- ⊕ Abstract classes and methods

Interfaces

⊕ **Collections**

- ⊕ ArrayList
- ⊕ HashMap
- ⊕ Iterator
- ⊕ Vector
- ⊕ Enumeration
- ⊕ Hashtable

⊕ **Exceptions**

- ⊕ Exception types
 - ⊕ Exception Hierarchy
 - ⊕ Catching exceptions
 - ⊕ Throwing exceptions
 - ⊕ Defining exceptions
- Common exceptions and errors

⊕ **Streams**

- ⊕ Stream types
- ⊕ Character streams
- ⊕ Byte streams
- ⊕ Filter streams
- ⊕ Object Serialization



Any questions?