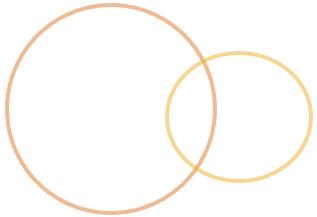
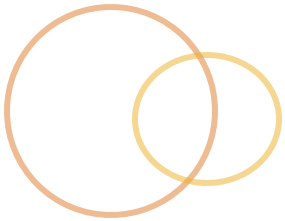
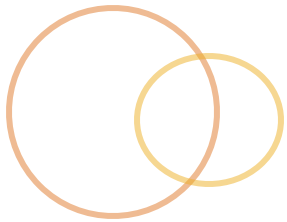
What's New in Java





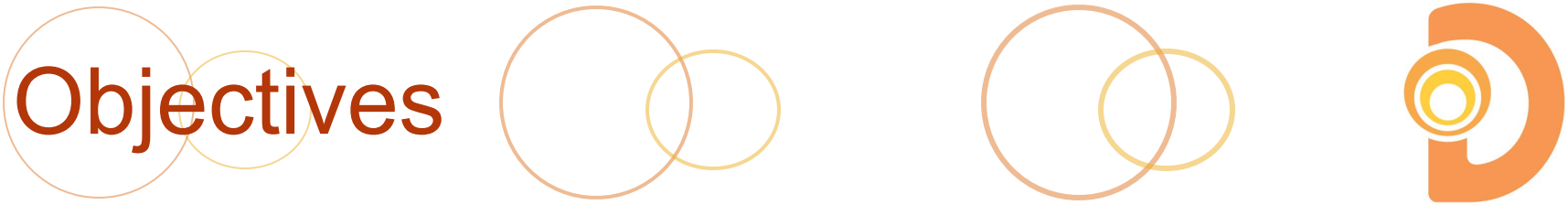
Presentation Topics



In this presentation, we will cover:

- Language Features
- API Additions

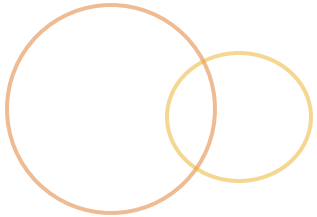
Objectives

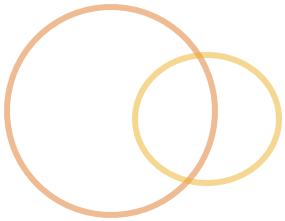


When we are done, you should be:

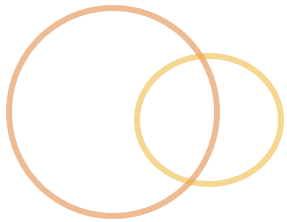
- Familiar with current state of Java
- Aware of new features

Language Features





Literal Improvements



Binary literals

```
int mask = 0b101010101010;
```

With underscores for clarity

```
int mask = 0b1010_1010_1010;
long big = 9_223_783_036_967_937L;
```

Simplification of Strings

- Strings are constants - And - treated like primitives (at least from a coding perspective)
- But until Java 7 – weren't supported in `switch` statements

```
String a = "Hello";
 . . .
switch(a) {
  case "hello":
  case "Hello":
  case "HELLO"
   //do something
  break;
  default:
  //do something else
  break;
}
```

Simplification of Generics

- Current way of declaring and initializing a type-safe collection

```
List<String> list = new ArrayList<String>();
```

- Compiler should be “smart enough” to infer type from declaration

- New syntax

```
List<String> list = new ArrayList<>();
```

Simplification of try/catch

- There's a lot of try/catch/finally boiler-plate code out there
- Why not let the compiler generate it for you?

```
FileInputStream fis;
try {
  fis = new FileInputStream("/tmp/myfile.txt");
  …
} catch(IOException ioe) {
  …
} finally {
  fis.close()
}
```

New way

```
try (InputStream fis =new FileInputStream("/tmp/myfile.txt")) {
  …
}
```

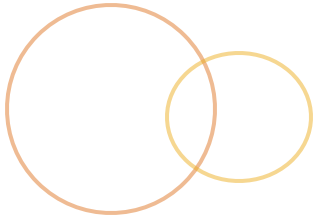
Simplification of try/catch

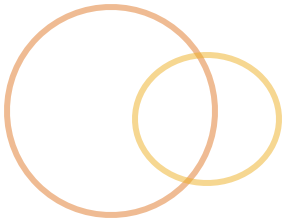
- Do we really need all those catches?

```
try {
   ...
} catch(ClassNotFoundException cnfe) {
   doSomethingClever(cnfe);
   throw cnfe;
} catch(InstantiationException ie) {
   log(ie);
   throw ie;
} catch(NoSuchMethodExceptic
   log(nsme);
   throw nsme;
} catch(InvocationTargetExce
   log(ite);
   throw ite;
}
```

```
try {
   ...
} catch (ClassCastException e) {
   doSomethingClever(e);
   throw e;
} catch(InstantiationException |
     NoSuchMethodException |
     InvocationTargetException e) {
   log(e);
   throw e;
}
```

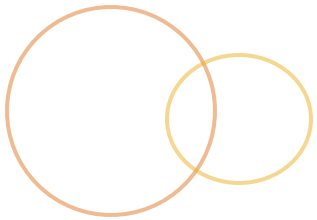
Strings

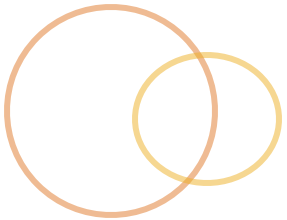




- `java.lang.String`
  - Better support for `CharSequence`
  - Built-in support for `format`
  - `isEmpty()`
- `java.lang.StringBuilder`
  - More efficient implementation of `StringBuffer`
  - Mutable String

Threads

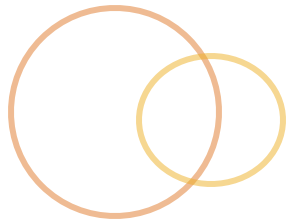




New methods in Thread

- `getStackTrace`
- `getAllStackTraces`
- `getId`
- `getState`
- `set/getDefaultExceptionHandler`
- `set/getUncaughtExceptionHandler`

java.util.Arrays



New methods in `Arrays` to provide same functionality `Arrays` have in other languages

- `deepHashCode()`
- `deepEquals`
- `toString`
- `deepToString`

java.util.Collections

New methods in Collections:

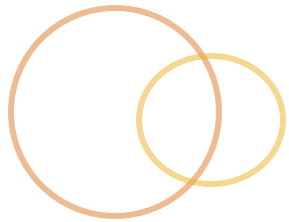
- checkCollection/Set/List…
- emptySet/List/Map
- reverseOrder
- frequency
- disjoint
- newSetFromMap
- asLifoQueue

Collections Framework

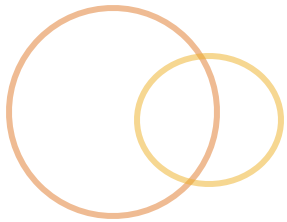
New set of collection interfaces

- `Deque / BlockingDeque` – double ended queue
  - Head operations: add / get / remove / peek
  - Tail operations: add / get / remove / peek
- `BlockingDeque` – deque with blocking
- `NavigableSet / NavigableMap / ConcurrentNavigableMap`
  - Sorted Collection with better navigation methods
  - `Lower, floor, ceiling, higher`
  - `headSet/headMap, tailSet/tailMap, subSet/subMap`

Collections Framework



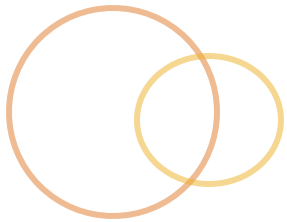
- New concrete implementations
  - `ArrayDeque`
  - `ConcurrentSkipListSet`
  - `ConcurrentSkipListMap`
  - `LinkedBlockingDeque`
- A Skip List
  - Data structure for storing a sorted list
  - Uses a hierarchy of linked lists
  - Lookup is log n



New classes in 1.6

- java.net.CookieManager
- java.net.CookiePolicy
- java.net.CookieStore
- java.net.HttpCookie
- java.net.IDN
- java.net.InMemoryCookieStore
- java.net.InterfaceAddress
- java.net.NetworkInterface

java.nio.file



New Package in 1.7

- Contains ~36 classes like
- `java.nio.file.Path`
- `java.nio.file.Filesystem`
- `java.nio.file.attribute`
- `java.nio.file.Files`
- `Java.nio.file.WatchService`

java.util.concurrent

13 New classes for 1.7

- java.util.concurrent.ConcurrentLinkedDeque
- java.util.concurrent.ForkJoinPool
- java.util.concurrent.ForkJoinTask
- java.util.concurrent.ForkJoinWorkerThread
- java.util.concurrent.LinkedTransferQueue
- java.util.concurrent.locks/AbstractQueuedLongSynchronizer
- java.util.concurrent.locks/AbstractQueuedSynchronizer
- java.util.concurrent.Phaser
- java.util.concurrent.RecursiveAction
- java.util.concurrent.RecursiveTask
- java.util.concurrent.ScheduledThreadPoolExecutor
- java.util.concurrent.ThreadLocalRandom
- java.util.concurrent.TransferQueue

Core Platform – Scripting

- JSR 223 – Scripting for the Java Platform
  - Java Applications can host scripting engines
  - Defined as a service
    - Scripting engines can be discovered through “service discovery” mechanism
    - Scripting engine should be contained as a JAR
    - Includes Mozilla Rhino as its JavaScript engine
- Currently implementations include
  - AWK
  - BeanShell
  - FreeMarker
  - Groovy
  - JavaScript
  - Jython
  - Jruby
  - and more…

Scripting Example – Embedded JS

```
1   package examples.platform;
2
3   import ...
6
7   /**...*/
18  public class EmbeddedJSExample {
19
20    public static void main(String[] args) {
21      ScriptEngineManager factory = new ScriptEngineManager();
22      ScriptEngine engine = factory.getEngineByName("JavaScript");
23
24      try {
25        engine.eval("print('Hello Scripting World')");
26      } catch (ScriptException e) {
27        System.err.println("Error processing JS: " + e);
28
29      }
30    }
31  }
```

Core Platform – Scripting

- Scripting support is full featured
  - Embedded – like example
  - External file – load in file and have engine evaluate it
  - Variables, functions, and methods
  - Java constructs including classes and interfaces
- For more information see:
  - http://java.sun.com/developer/technicalArticles/J2SE/Desktop/scripting/
  - http://java.sun.com/javase/6/docs/technotes/guides/scripting/programmer_guide/index.html
  - http://jcp.org/en/jsr/detail?id=223