Core Java

8. Classes: Creation, Overloading, Autoboxing, Packages

Creation

- Objects are created through the *new* operator
- Objects are initialized by special functions called Constructors
- Constructors have the same name as the class
- They take arguments which are used to initialize per-object data
- Constructors are usually *public*

Constructors

- After a Constructor is called, an Object's other functions can be used
- Constructors can not be invoked directly
- Constructors don't have a return type
- Appropriate Constructors are called by invoking the new clause with arguments which match its signature
- Java does not have destructors

Finalize

- Java, however, has a method called *void finalize()*
- It is invoked when an object is about to be garbage collected
- There is no explicit way to invoke an object destruction, merely set it to *null*
- Calling *System.gc()* will invoke the garbage collector

Overloading

- Overloading a function requires writing multiple versions of functions
- All the versions have the same name
- The arguments are all that make a difference
- In case of automatic type-promotion, the appropriate function will be chosen
- *System.out.println()* is overloaded
- Constructors can be overloaded as well

Autoboxing

- In Java, it is possible to convert a primitive to it's equivalent Object
- All primitives can be converted to/fro objects
- Makes it easier to write code based on inheritance then overloading
- *toString()* method of each Object can be invoked

Packages

- Logically group classes into a package
- Package names are usually generic or vendor-specific
- Access rules apply to classes
- Makes it easy to distribute your applications with limited visibility: black box approach
- Declare Packages before compiling
- Import for using

Demonstration

- Compile and Execute a few programs

Questions?