Package
com.projectReflection

com.projectReflection
Class ClassReflection

```
java.lang.Object
    │
    +-com.projectReflection.ClassReflection
```

public class **ClassReflection**
extends java.lang.Object

This class allows for mapping of a Class.

Constructor Summary

public	[ClassReflection](#)()

Method Summary

void	[display](#)(java.lang.Class theClass) Displays a structural mapping of the class
java.util.List	[filterMethodsByParams](#)(java.util.List methods, java.lang.String[] params) Filters a list of methods by parameter type.
java.lang.reflect.Field	[getFieldByString](#)(java.lang.String fieldName, java.lang.Class theClass) Gets the field that matches the given field name exactly and exists in the given class.
java.util.List	[getFieldsByPattern](#)(java.util.regex.Pattern fieldName, java.lang.Class theClass) Gets a list of fields that match the given regular expression and exist in the given class.
java.util.List	[getMethodsByPattern](#)(java.util.regex.Pattern methodName, java.lang.Class theClass) Gets a list of methods that match the regular expression and exist in the given class.
java.util.List	[getMethodsByString](#)(java.lang.String methodName, java.lang.Class theClass) Gets a list of methods that match the specified method name in a class.

Methods inherited from class java.lang.Object

```
clone, equals, finalize, getClass, hashCode, notify, notifyAll, registerNatives,
toString, wait, wait, wait
```

Constructors

ClassReflection

public **ClassReflection**()

(continued on next page)

Methods

getMethodsByPattern

```
public java.util.List getMethodsByPattern(java.util.regex.Pattern methodName,
        java.lang.Class theClass)
```

Gets a list of methods that match the regular expression and exist in the given class. It loops through the methods of the class and adds each method that matches the regular expression to a list. That list is returned. If no matches are found, the list is empty.

Parameters:
    `methodName` - The regular expression used for matching the method names
    `theClass` - The class to search in for the method name

Returns:
    a list of methods that match the regular expression
    the list is empty if no matches are found

getMethodsByString

```
public java.util.List getMethodsByString(java.lang.String methodName,
        java.lang.Class theClass)
```

Gets a list of methods that match the specified method name in a class. It generates a Pattern to exactly match the string calls getMethodsByPattern(Pattern, Class)

Parameters:
    `methodName` - The method name to search for
    `theClass` - The class to search in for the method name

Returns:
    a list of methods that match the exact method name
    the list is empty if no matches are found

filterMethodsByParams

```
public java.util.List filterMethodsByParams(java.util.List methods,
        java.lang.String[] params)
```

Filters a list of methods by parameter type.

Parameters:
    `methods` - the list of methods to filter
    `params` - the array of strings that contain the names of the classes that the parameters should match - names can be "simple names" (for example: "Bar") or fully qualified names (for example: "com.foo.bar")

Returns:
    list of filtered methods
    empty list of no matches are found

getFieldsByPattern

```
public java.util.List getFieldsByPattern(java.util.regex.Pattern fieldName,
        java.lang.Class theClass)
```

Gets a list of fields that match the given regular expression and exist in the given class. It loops through the fields of the class and checks for matches. When a match is found, it is added to a list of fields. That list is returned. If no matches are found, the list is empty.

Parameters:

fieldName - The name of the field to search for

theClass - The class to search in for the field

Returns:

list of fields that match the regular expression

list is empty if no matches were found

getFieldByString

```
public java.lang.reflect.Field getFieldByString(java.lang.String fieldName,
        java.lang.Class theClass)
```

Gets the field that matches the given field name exactly and exists in the given class.

Parameters:

fieldName - The name of the field to search for

theClass - The class to search in for the field

Returns:

the field if a match is found

otherwise, null

display

```
public void display(java.lang.Class theClass)
```

Displays a structural mapping of the class

Parameters:

theClass - the class being displayed

com.projectReflection
Class ProjectReflection

```
java.lang.Object
   │
   +-com.projectReflection.ProjectReflection
```

public class **ProjectReflection**
extends java.lang.Object

This class allows for the mapping of a project.

Field Summary

public static final	[CR](#)
private static final	[DOT](#) Value: **46**
private static final	[SLASH](#) Value: **47**

Constructor Summary

public	[ProjectReflection](#)()

Method Summary

boolean	[classExists](#)(java.lang.String className, java.util.List classes) Checks to see if a specific class exists in a list of classes.
boolean	[classExists](#)(java.lang.String className, java.lang.String packageName) Checks to see if a specific class exists in a package.
void	[displayAll](#)(java.util.List classes) Displays a structural mapping of each class in a list.
java.util.List	[getAllLoadedClasses](#)() Gets all classes that are loaded.
java.lang.Class	[getClass](#)(java.lang.String className, java.util.List classes) Gets a specified class from a list of classes.
java.lang.Class	[getClass](#)(java.lang.String className, java.lang.String packageName) Gets a specified class from a package.
java.util.List	[getLoadedClassesFromFile](#)(java.io.File file) Gets all loaded classes in a file.
java.util.List	[getLoadedClassesFromFile](#)(java.lang.String fileName) Gets all loaded classes from a file (or directory).

java.util.List	[getLoadedClassesFromJarFile](java.util.jar.JarFile jarFile) Gets all loaded classes in a jar file.
java.util.List	[getLoadedClassesFromPackage](java.lang.String scannedPackage) Generates a List of classes that exist within a specified package.
java.lang.Class	[getLoadedClassFromDotClassFile](java.io.File file) Gets the loaded classes in a .class file.

Methods inherited from class `java.lang.Object`

```
clone, equals, finalize, getClass, hashCode, notify, notifyAll, registerNatives,
toString, wait, wait, wait
```

Fields

DOT

```
private static final char DOT
```

Constant value: **46**

SLASH

```
private static final char SLASH
```

Constant value: **47**

CR

```
public static final com.projectReflection.ClassReflection CR
```

Constructors

ProjectReflection

```
public ProjectReflection()
```

Methods

getAllLoadedClasses

```
public java.util.List getAllLoadedClasses()
```

Gets all classes that are loaded.

Returns:
A list of all loaded classes

getLoadedClassesFromFile

```
public java.util.List getLoadedClassesFromFile(java.lang.String fileName)
```

Gets all loaded classes from a file (or directory). Calls getLoadedClassesFromFile(File)

Parameters:
fileName - The name of the file (or directory) to search in

Returns:
list of loaded classes found
empty list if none are found

getLoadedClassesFromFile

```
public java.util.List getLoadedClassesFromFile(java.io.File file)
```

Gets all loaded classes in a file.

Parameters:
file - The file to search for loaded classes in

Returns:
list of loaded classes from the jar file
empty list if no loaded classes are found

getLoadedClassesFromJarFile

```
public java.util.List getLoadedClassesFromJarFile(java.util.jar.JarFile jarFile)
```

Gets all loaded classes in a jar file.

Parameters:
jarFile - The jar file to search for loaded classes in

Returns:
list of loaded classes in the jar file
empty list if none are found

getLoadedClassFromDotClassFile

```
public java.lang.Class getLoadedClassFromDotClassFile(java.io.File file)
```

Gets the loaded classes in a .class file.

Parameters:
file - The file to search for loaded a class in

Returns:
the class in the .class file if it is loaded
null if the class is not loaded

getLoadedClassesFromPackage

```
public java.util.List getLoadedClassesFromPackage(java.lang.String scannedPackage)
```

Generates a List of classes that exist within a specified package.

Parameters:
scannedPackage - Specifies the package to scan for classes

(continued from last page)

Returns:
>   A list of classes found

---

classExists

```
public boolean classExists(java.lang.String className,
        java.util.List classes)
```

Checks to see if a specific class exists in a list of classes.

Parameters:
>   `className` - The specified name of the class being checked for existence - this can be the *simple name* (for example: "Bar") or it can be the *fully qualified name* (for example: "com.foo.Bar")
>   `classes` - The list of classes being searched

Returns:
>   true if the class is found
>   false if the class is not found

---

classExists

```
public boolean classExists(java.lang.String className,
        java.lang.String packageName)
```

Checks to see if a specific class exists in a package. This method calls classExists(String, List) after generating a list of classes found in the package specified by "packageName".

Parameters:
>   `className` - The specified name of the class being checked for existence - this can be the *simple name* (for example: "Bar") or it can be the *fully qualified name* (for example: "com.foo.Bar")
>   `packageName` - Specifies the package name to use when generating a list of classes

Returns:
>   true if the class is found
>   false if the class is not found

---

getClass

```
public java.lang.Class getClass(java.lang.String className,
        java.util.List classes)
```

Gets a specified class from a list of classes. It compares the class name to the classes and returns the class if it is found. Otherwise, it returns null.

Parameters:
>   `className` - The specified name of the class being checked for existence - this can be the *simple name* (for example: "Bar") or it can be the *fully qualified name* (for example: "com.foo.Bar")
>   `classes` - The list of classes being searched

Returns:
>   the class if exists
>   null if the class does not exist

---

getClass

```
public java.lang.Class getClass(java.lang.String className,
        java.lang.String packageName)
```

Gets a specified class from a package. It creates a list of classes and calls getClass(String, List) which returns the class if it is found. Otherwise, it returns null.

(continued from last page)

Parameters:
    `className` - The specified name of the class being checked for existence - this can be the *simple name* (for example: "Bar") or it can be the *fully qualified name* (for example: "com.foo.Bar")
    `packageName` - The specified package to be used when creating the list of classes

Returns:
    the class if exists
    null if the class does not exist

---

displayAll

```
public void displayAll(java.util.List classes)
```

Displays a structural mapping of each class in a list.

Parameters:
    `classes` - list of classes that are to be displayed