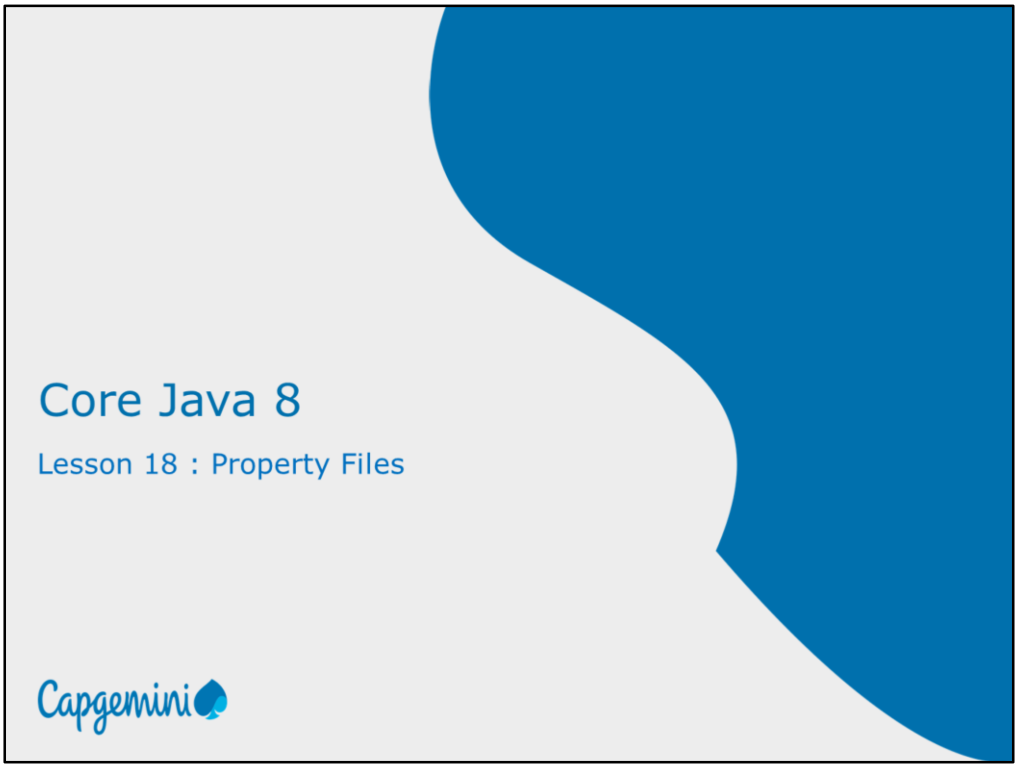
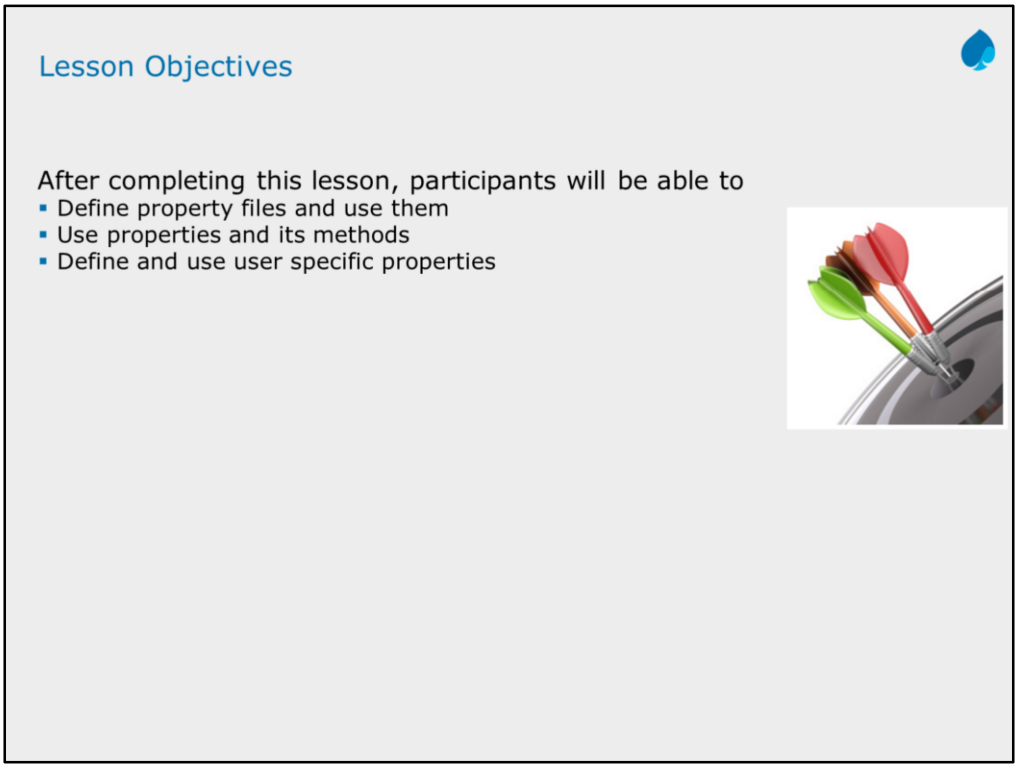


Core Java 8

Lesson 18 : Property Files

Capgemini

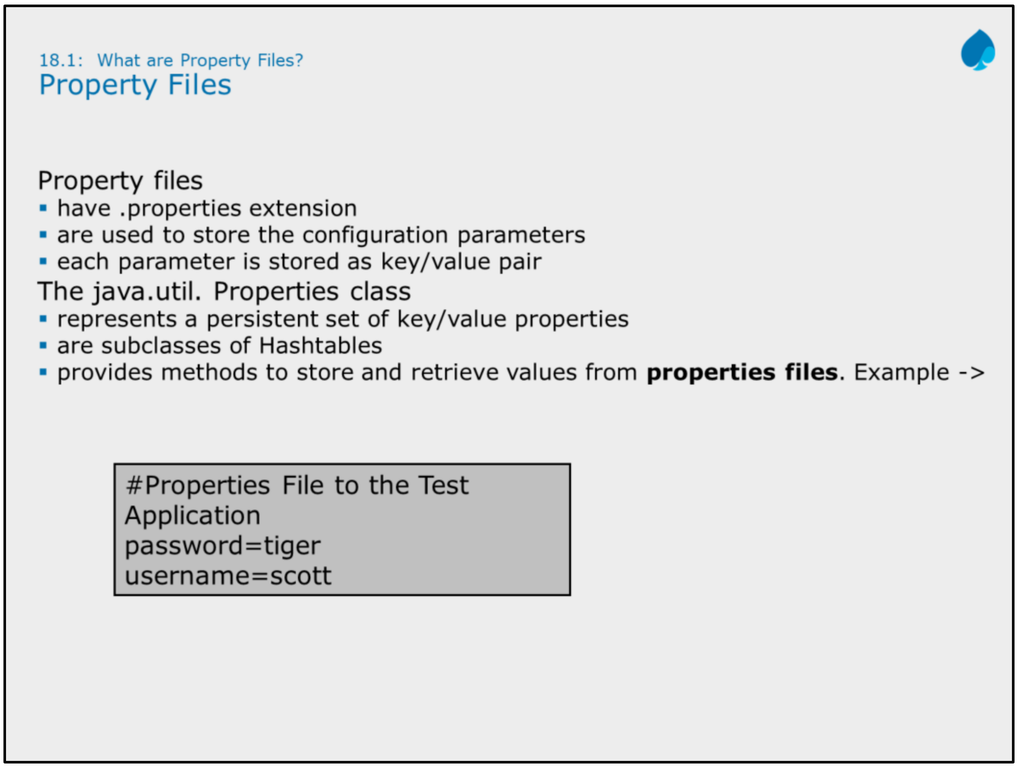


Lesson Objectives

After completing this lesson, participants will be able to
- Define property files and use them
- Use properties and its methods
- Define and use user specific properties

This lesson covers the usage of Property files in your application. It explains how to create user specific property file. The important Properties class methods are also explained

Lesson outline:



18.1: What are Property Files?

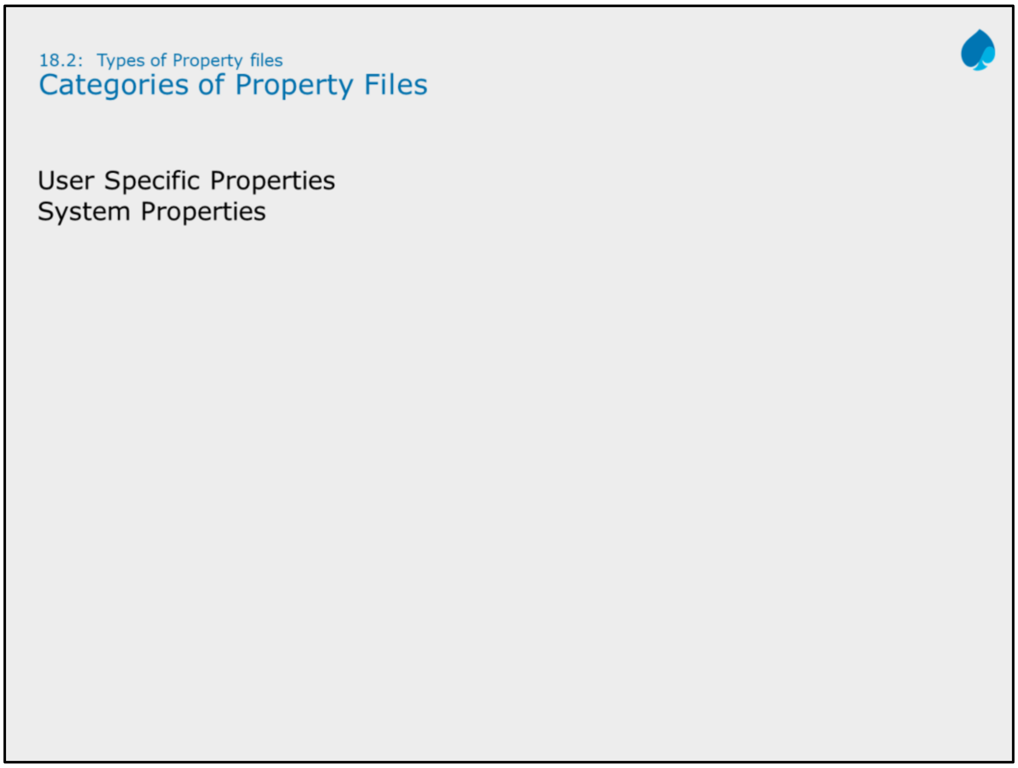
Property Files

Property files
- have .properties extension
- are used to store the configuration parameters
- each parameter is stored as key/value pair

The java.util. Properties class
- represents a persistent set of key/value properties
- are subclasses of Hashtables
- provides methods to store and retrieve values from **properties files**. Example ->

```
#Properties File to the Test
Application
password=tiger
username=scott
```

Property Files:
Property files come with .properties extension and are used to store the configuration parameters or setting. Each parameter is stored as a pair of strings, one storing the name of the parameter (called the key), and the other storing the value.
java.util.Properties represents a persistent set of properties, ie a “key=value” pair. Each key and its corresponding value in the property list is a string. Properties are subclasses of Hashtables that can be backed to disk in human-readable format. You lookup by property name and get a value.
The Properties class provides methods to store and retrieve values from properties files.
The following are some of the points to be noted about Properties file:
Comments begin with #.
The keywords can contain dots and underscores but not spaces or =. You can use _ (underscore) in key names to represent a space.
The values can contain dots, underscores, spaces, and =.

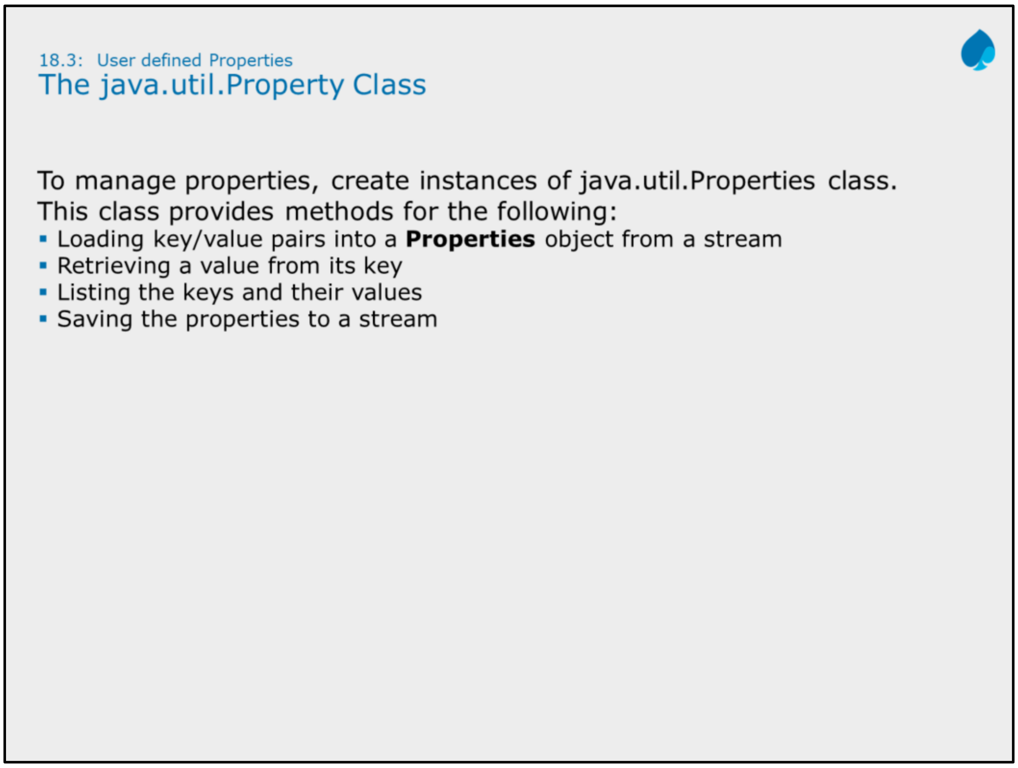

18.2: Types of Property files
Categories of Property Files

User Specific Properties
System Properties

Categories of Property Files:

User Specific Properties
These properties are part of the Application.properties containing a key value pair, which can be mentioned by the program in run. User-specific properties are generally used for configuring the application.
Our focus in this lesson will be on user specific property files.

System Properties
The Java platform itself uses a Properties object to maintain its own configuration. The System class maintains a Properties object that describes the configuration of the current working environment. System properties include information about the current user, the current version of the Java runtime, and the character used to separate components of a file path name.
You may read up on System properties in Appendix-A.

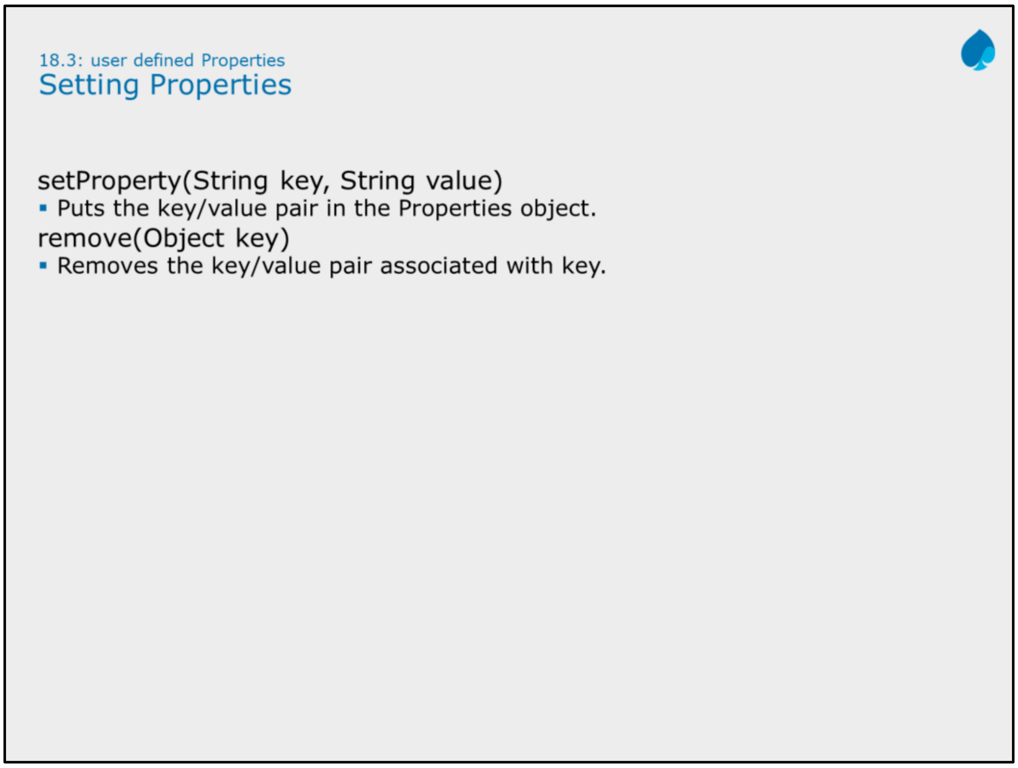


18.3: User defined Properties

The java.util.Property Class

To manage properties, create instances of java.util.Properties class.
This class provides methods for the following:
- Loading key/value pairs into a **Properties** object from a stream
- Retrieving a value from its key
- Listing the keys and their values
- Saving the properties to a stream

The java.util.Property Class
Some of the widely used methods of the java.util.Properties class:-
load
public synchronized void load(InputStream inStream) throws IOException : reads a property list (key and element pairs) from the input stream.
getProperty
public String getProperty(String key) : Searches for the property with the specified key in this property list. If the key is not found in this property list, the default property list, and its defaults, recursively, are then checked. The method returns null if the property is not found.
list
public void list(PrintStream out) : Prints this property list out to the specified output stream. This method is useful for debugging.
save
public synchronized void save(OutputStream out, String header) : Calls the store(OutputStream out, String header) method and suppresses IOExceptions that were thrown.


18.3: user defined Properties

Setting Properties

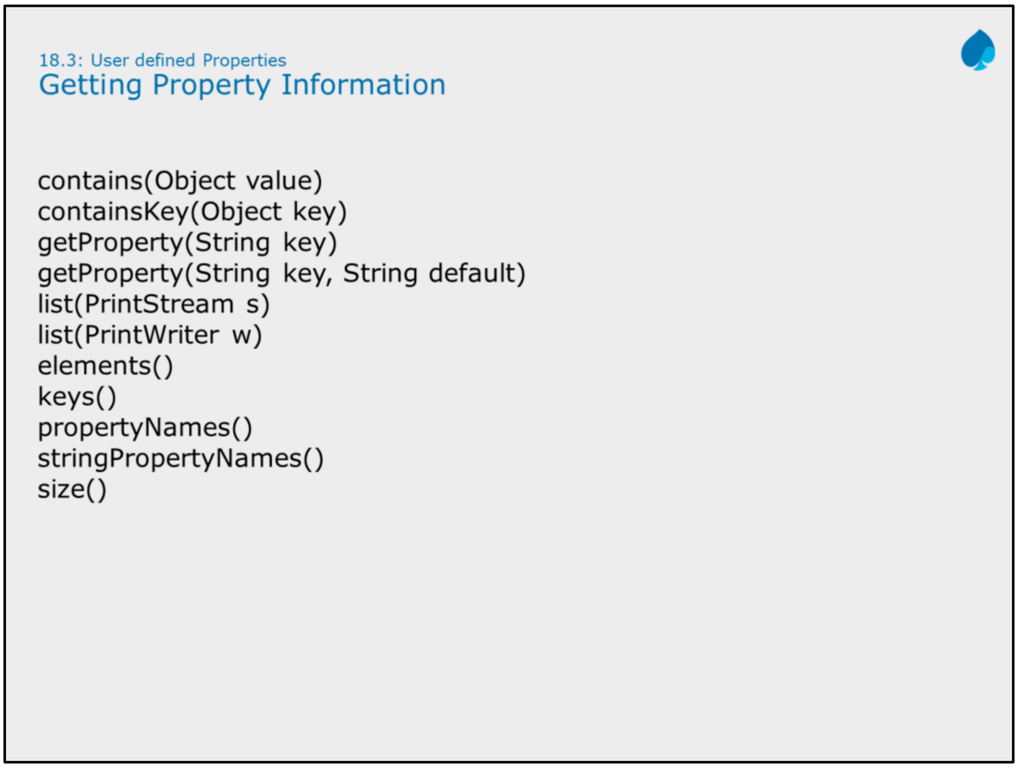
setProperty(String key, String value)
- Puts the key/value pair in the Properties object.

remove(Object key)
- Removes the key/value pair associated with key.

Setting Properties:
A user's interaction with an application during its execution may impact property settings. These changes should be reflected in the Properties object so that they are saved when the application exits (and calls the store method).

The following methods change the properties in a Properties object:
setProperty(String key, String value)
Puts the key/value pair in the Properties object.
remove(Object key)
Removes the key/value pair associated with key.

Note: Some of the methods described above are defined in Hashtable, and thus, they accept key and value argument types other than String. Always use Strings for keys and values, even if the method allows other types. Also, do not invoke Hashtable.set or Hastable.setAll on Properties objects; always use Properties.setProperty.



18.3: User defined Properties

Getting Property Information

contains(Object value)
containsKey(Object key)
getProperty(String key)
getProperty(String key, String default)
list(PrintStream s)
list(PrintWriter w)
elements()
keys()
propertyNames()
stringPropertyNames()
size()

Getting Property Information

contains(Object value) , containsKey(Object key)

This returns TRUE if the value or the key is in the Properties object. Properties inherits these methods from Hashtable. Thus, they accept Object arguments, but only String values should be used.

getProperty(String key) , getProperty(String key, String default)

This returns the value for the specified property. The second version provides for a default value. If the key is not found, the default is returned.

list(PrintStream s) , list(PrintWriter w)

This writes all of the properties to the specified stream or writer. This is useful for debugging.

elements() , keys() , propertyNames()

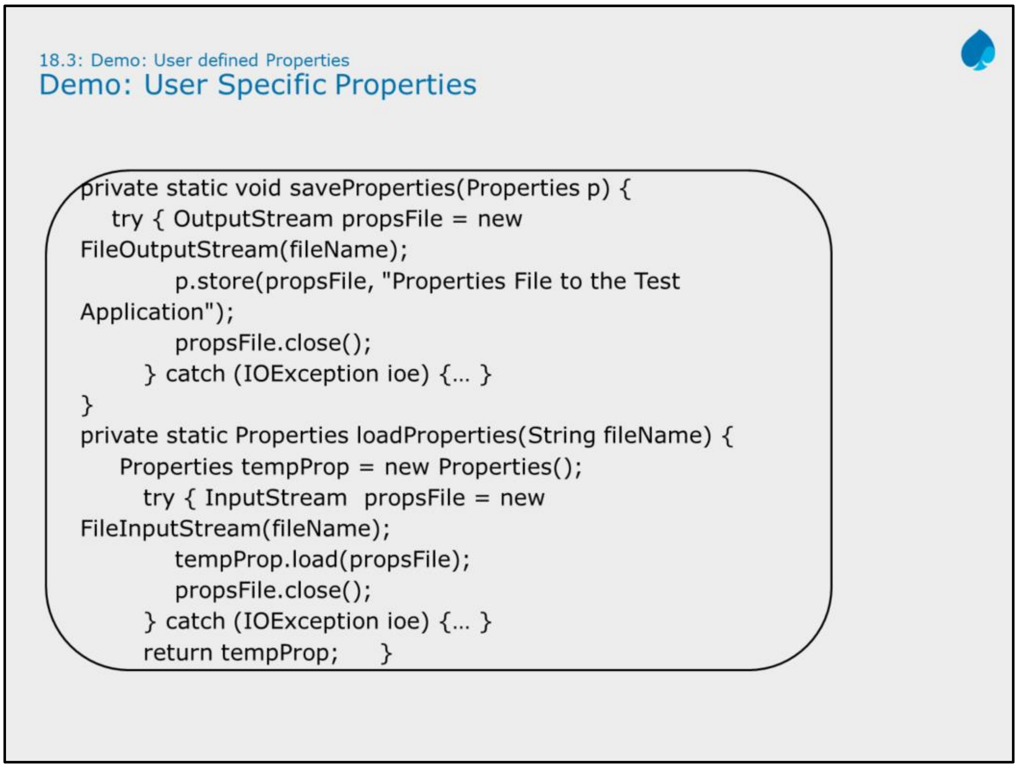
This returns an Enumeration containing the keys or values (as indicated by the method name) contained in the Properties object. The keys method only returns the keys for the object itself; the propertyNames method returns the keys for default properties as well.

string PropertyNames()

This functions like propertyNames, but returns a Set<String>, and only returns names of properties where both key and value are strings. Note that the Set object is not backed by the Properties object, so changes in one do not affect the other.

size()

This returns the current number of key/value pairs.

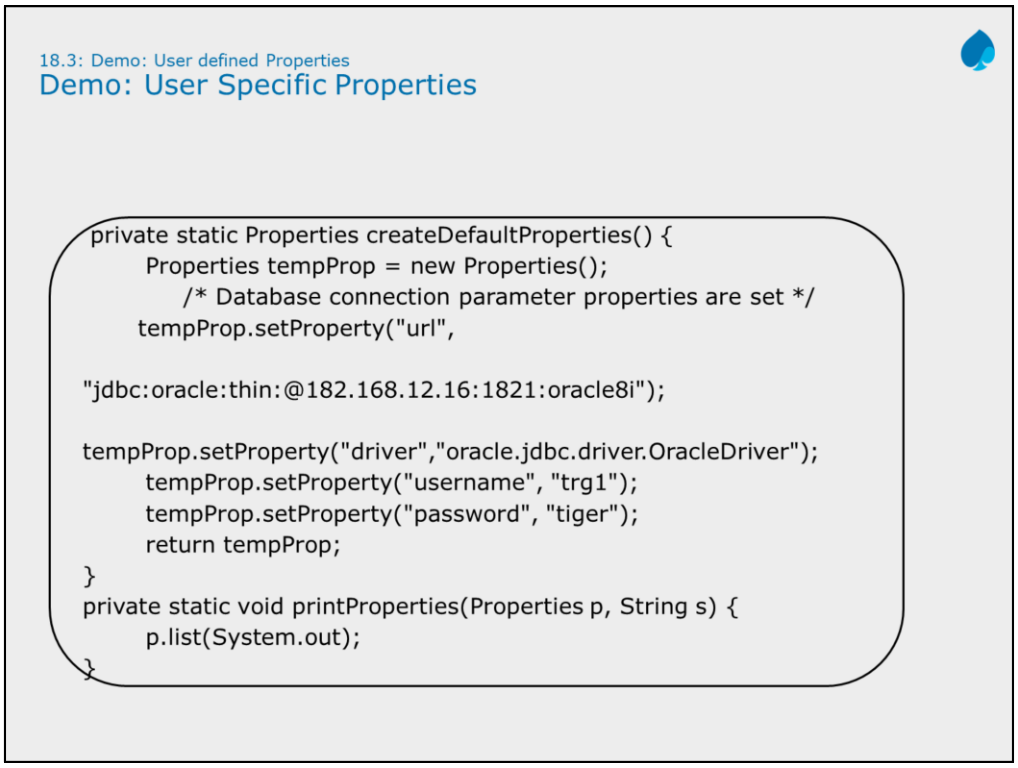


18.3: Demo: User defined Properties

Demo: User Specific Properties

```
private static void saveProperties(Properties p) {
   try { OutputStream propsFile = new
FileOutputStream(fileName);
            p.store(propsFile, "Properties File to the Test
Application");
            propsFile.close();
         } catch (IOException ioe) {... }
}
private static Properties loadProperties(String fileName) {
     Properties tempProp = new Properties();
        try { InputStream  propsFile = new
FileInputStream(fileName);
            tempProp.load(propsFile);
            propsFile.close();
         } catch (IOException ioe) {... }
         return tempProp;     }
```

Add the notes here.



18.3: Demo: User defined Properties

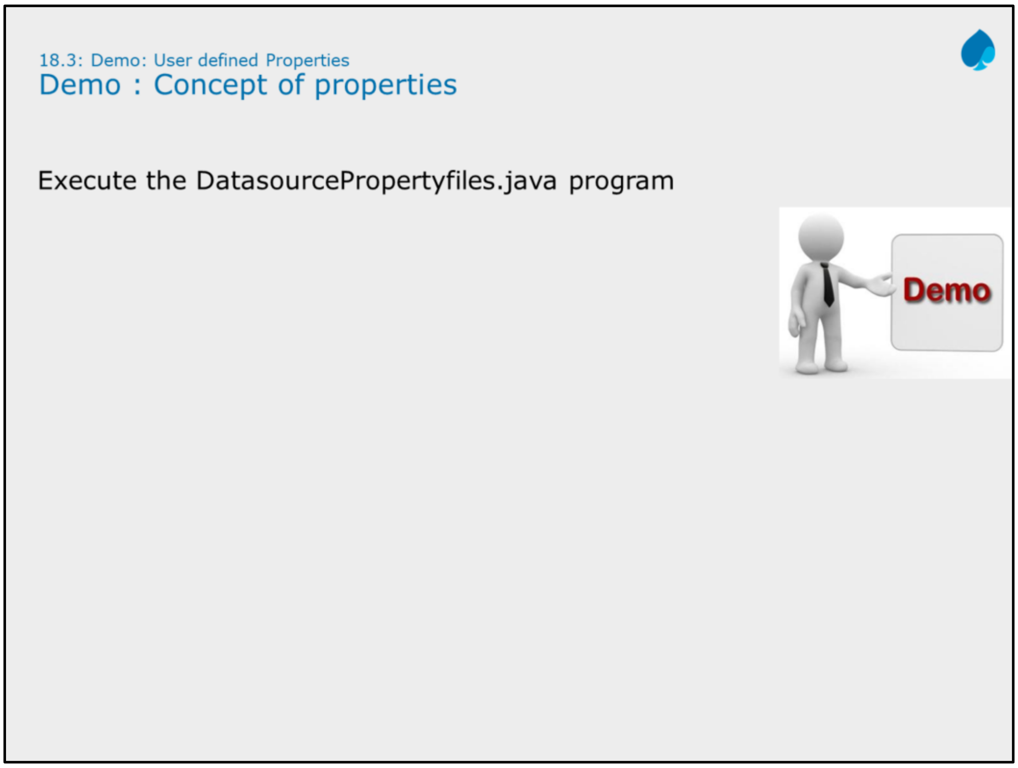
Demo: User Specific Properties

```
private static Properties createDefaultProperties() {
       Properties tempProp = new Properties();
           /* Database connection parameter properties are set */
      tempProp.setProperty("url",

"jdbc:oracle:thin:@182.168.12.16:1821:oracle8i");

tempProp.setProperty("driver","oracle.jdbc.driver.OracleDriver");
       tempProp.setProperty("username", "trg1");
       tempProp.setProperty("password", "tiger");
       return tempProp;
}
private static void printProperties(Properties p, String s) {
       p.list(System.out);
}
```

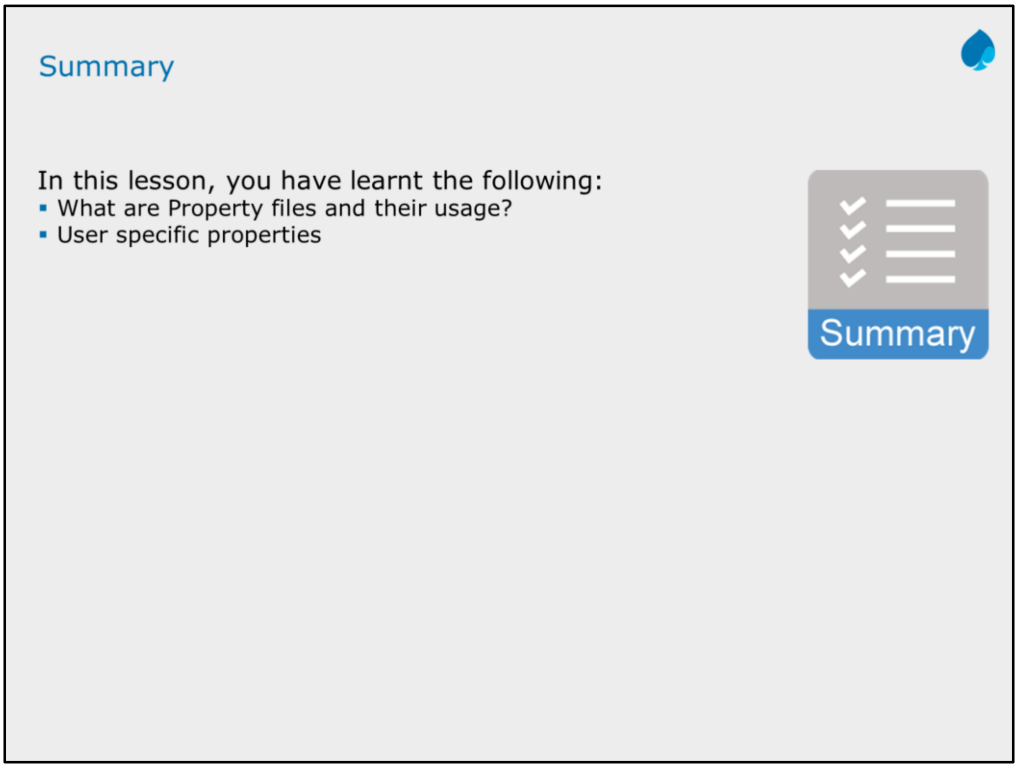
Add the notes here.



18.3: Demo: User defined Properties

Demo : Concept of properties

Execute the DatasourcePropertyfiles.java program



Summary

In this lesson, you have learnt the following:

- What are Property files and their usage?
- User specific properties

Add the notes here.

Review Question

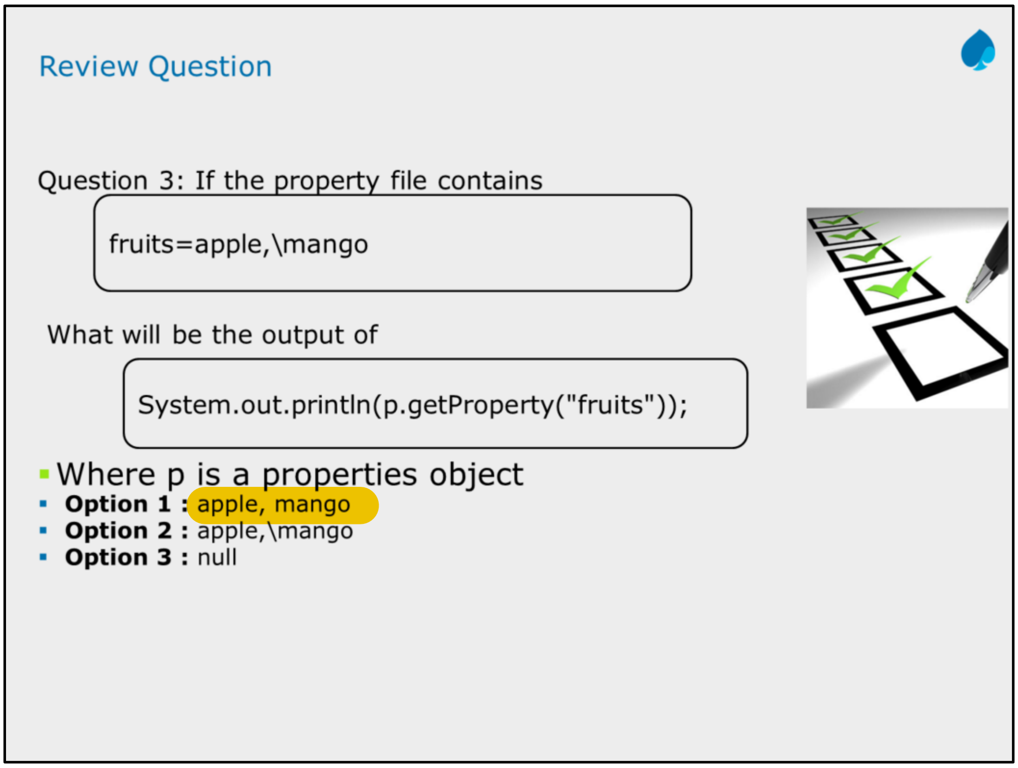
Question 1: load(_______________) throws IOException

- **Option 1 :** InputStream
- **Option 2 :** OutputStream

Question 2: Is this a valid key value pair?

fruit apple

- True/False.



Review Question

Question 3: If the property file contains

```
fruits=apple,\mango
```

What will be the output of

```
System.out.println(p.getProperty("fruits"));
```

- Where p is a properties object
  - **Option 1 :** [✓ apple, mango]
  - **Option 2 :** apple,\mango
  - **Option 3 :** null