

A decorative graphic on the right side of the page. It features three sets of concentric circles in shades of blue. The top set is the largest, the middle set is medium-sized, and the bottom set is the smallest. Thin blue lines extend from the top-left and top-right towards the circles, and a thicker blue line extends from the bottom-right towards the largest circle.

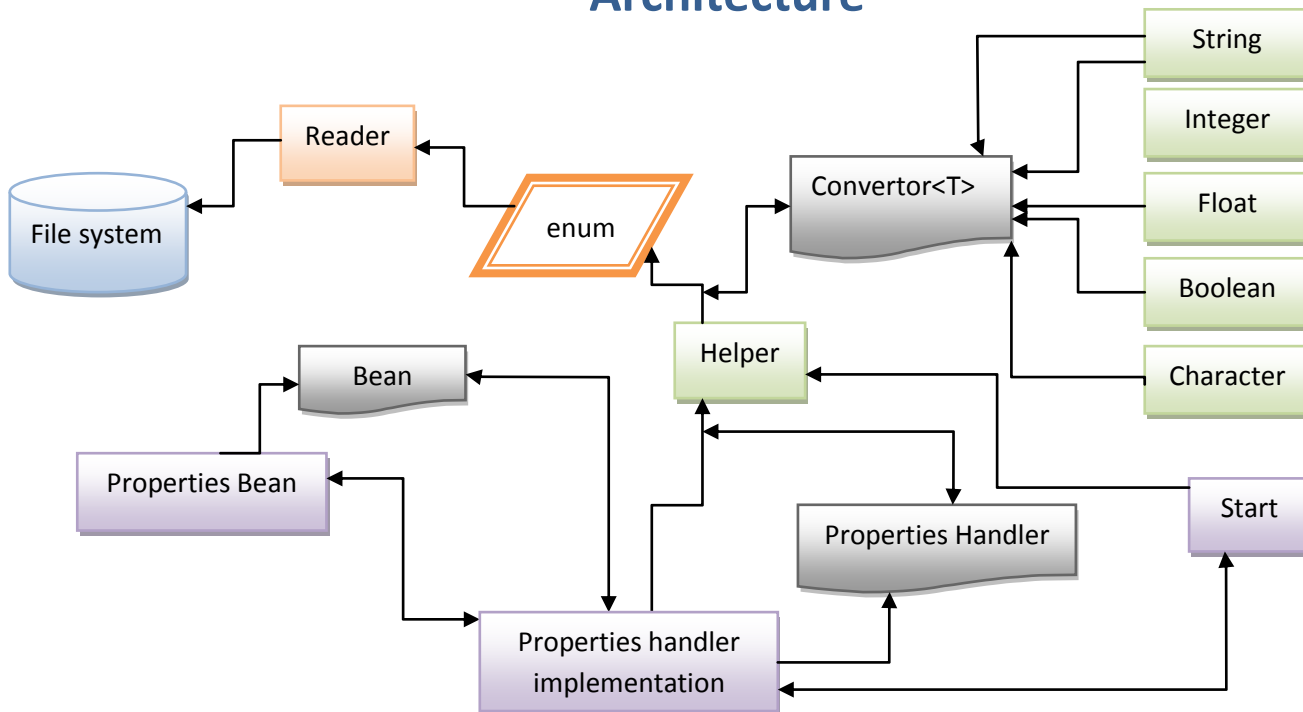
JConfigReader

Easier way to read config files

This document illustrates the usage of JConfigReader, a utility that can be use in big projects to upload/modify the configurations such as properties, XML, and ini files in a safer and easier manner.

Sunny Jain
6/2/2010

Architecture



Class contains the root implementation. One using JConfigReader will never use this class directly.

Interfaces used to implement the basic functionality.

Classed we can use to implement the upload the configurations.

Classes one has to write in order to implement JConfigReader.

Algorithm

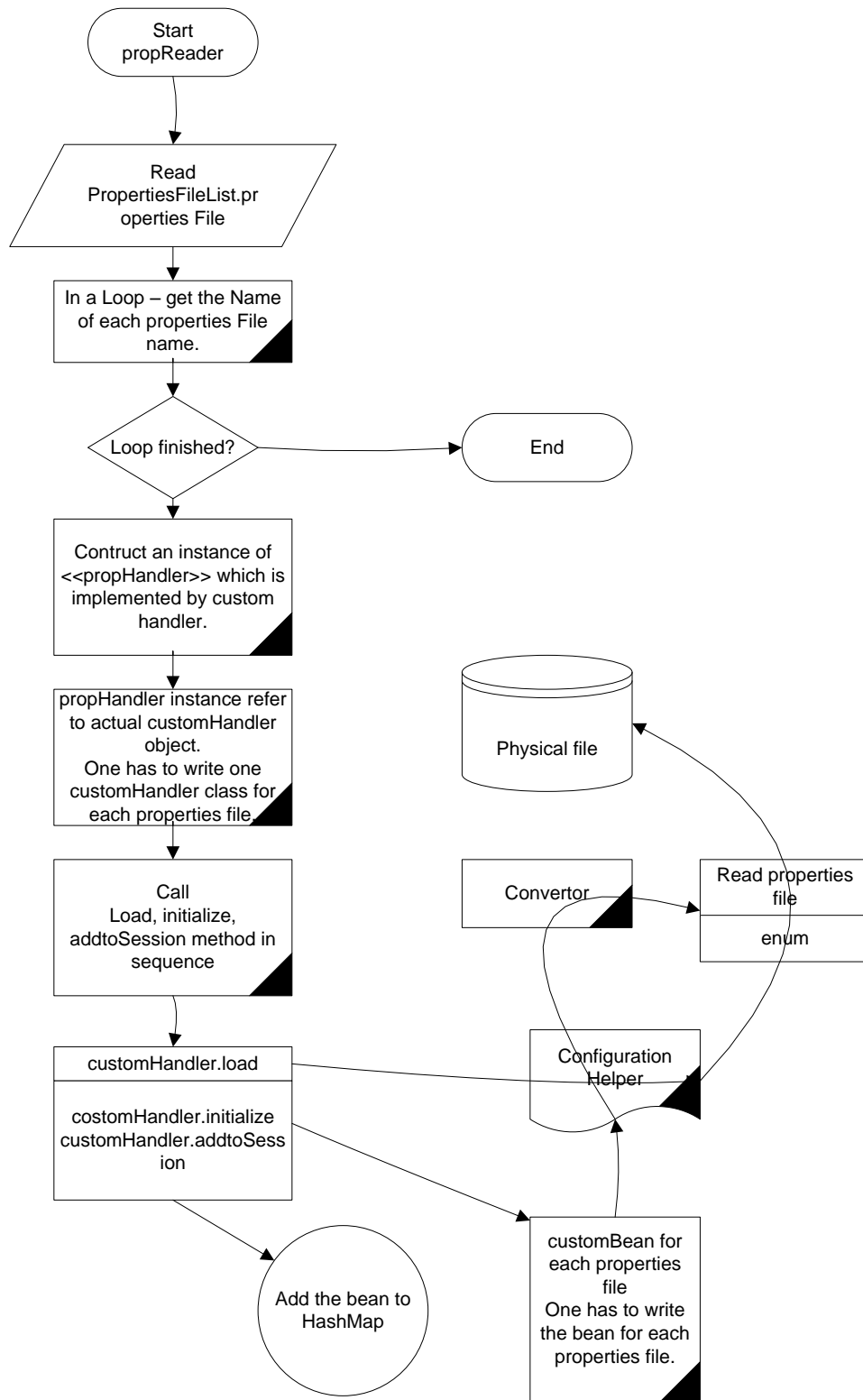


Illustration – How to add a new property file?

3 step process –

- 1) Add the property file name in Properties File.

For example- I want to add a new property file name 'sunny.properties'.

We have to increase the key value and add the corresponding property file name.

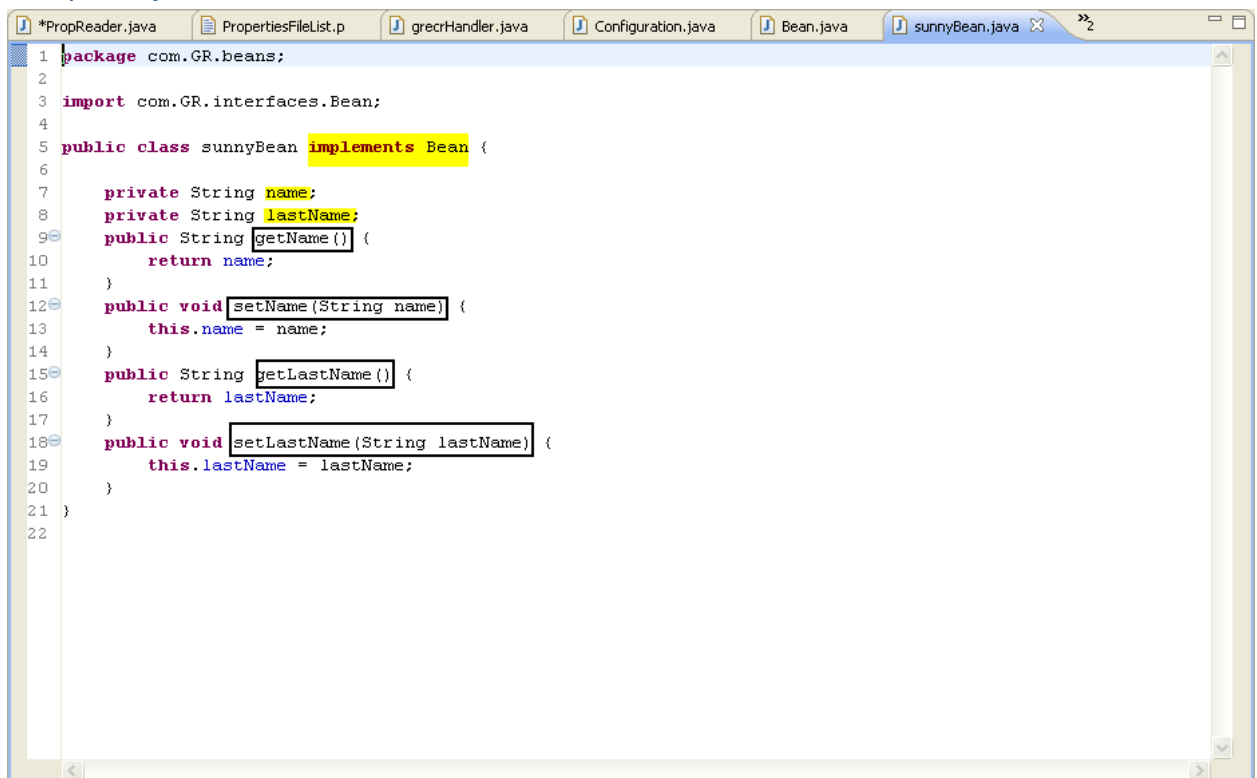
```
Field1=gregr.properties  
Field2=sunny.properties
```

- 2) Create sunnyBean.java.

We have to make sure that name of Bean Class is – name of property file + Bean.java

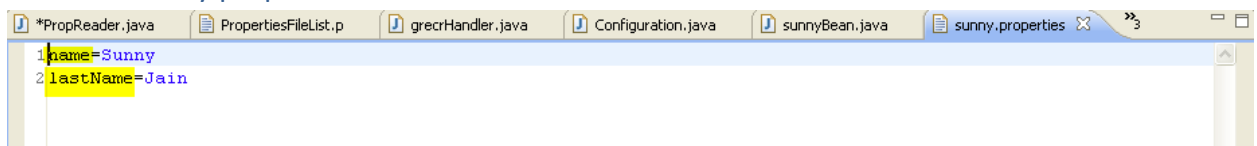
So “ **sunny+Bean.java = sunnyBean.java**”

sunnyBean.java looks like this –



```
1 package com.GR.beans;  
2  
3 import com.GR.interfaces.Bean;  
4  
5 public class sunnyBean implements Bean {  
6  
7     private String name;  
8     private String lastName;  
9     public String getName() {  
10         return name;  
11     }  
12     public void setName(String name) {  
13         this.name = name;  
14     }  
15     public String getLastName() {  
16         return lastName;  
17     }  
18     public void setLastName(String lastName) {  
19         this.lastName = lastName;  
20     }  
21 }  
22
```

Where as sunny.properties file looks like this –



```
1 name=Sunny  
2 lastName=Jain
```

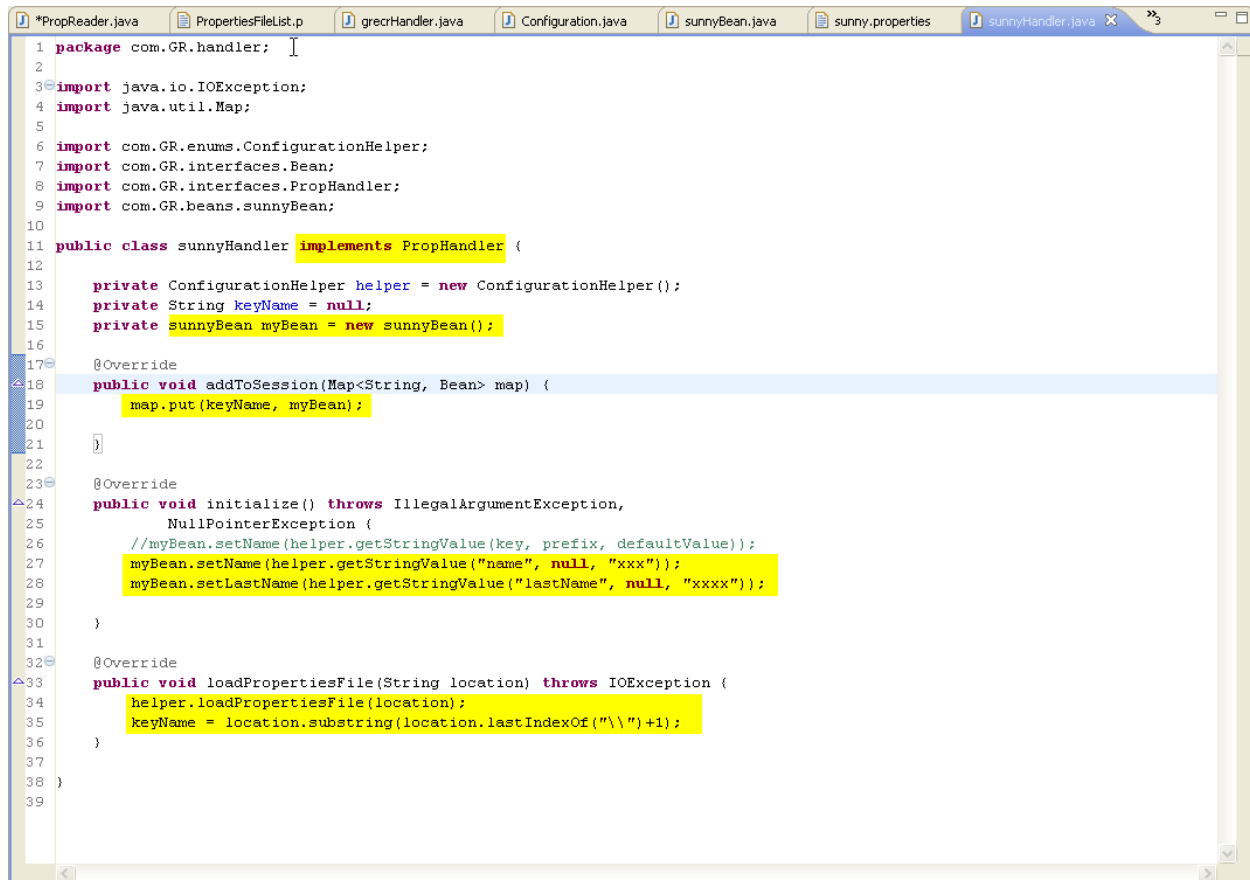
Note – We must have getter and setter corresponding to each field in properties file.

Step3: Create sunnyHandler.java

We have to make sure that name of Bean Class is – name of property file + Handler.java

So “sunny+Handler.java = sunnyHandler.java”

sunnyHandler.java looks like this –



```
1 package com.GR.handler;
2
3 import java.io.IOException;
4 import java.util.Map;
5
6 import com.GR.enums.ConfigurationHelper;
7 import com.GR.interfaces.Bean;
8 import com.GR.interfaces.PropHandler;
9 import com.GR.beans.sunnyBean;
10
11 public class sunnyHandler implements PropHandler {
12
13     private ConfigurationHelper helper = new ConfigurationHelper();
14     private String keyName = null;
15     private sunnyBean myBean = new sunnyBean();
16
17     @Override
18     public void addToSession(Map<String, Bean> map) {
19         map.put(keyName, myBean);
20     }
21
22     @Override
23     public void initialize() throws IllegalArgumentException,
24         NullPointerException {
25         //myBean.setName(helper.getStringValue(key, prefix, defaultValue));
26         myBean.setName(helper.getStringValue("name", null, "xxx"));
27         myBean.setLastName(helper.getStringValue("lastName", null, "xxxx"));
28     }
29
30     @Override
31     public void loadPropertiesFile(String location) throws IOException {
32         helper.loadPropertiesFile(location);
33         keyName = location.substring(location.lastIndexOf("\\")+1);
34     }
35
36 }
37
38
39
```

That’s all – you in this way you will add the new properties bean in session Map.