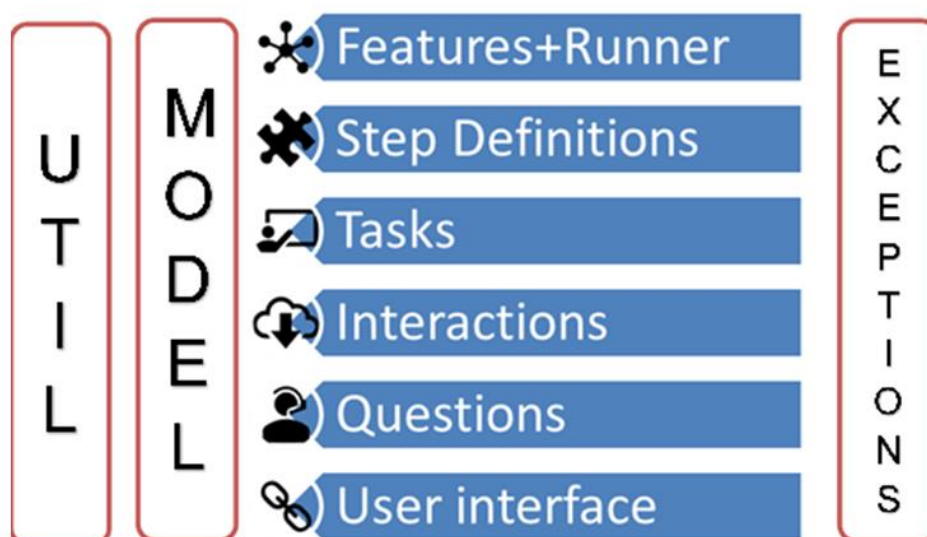


7

GUÍA – SCREENPLAY CON MYEXTRA

Nota: En el siguiente documento compartiremos una alternativa para el uso de la herramienta MyExtra bajo una estructura similar a la propuesta por el patrón Screenplay. Para la realización de la guía, sería conveniente estar familiarizado con el patrón screenplay original y su estructura.



Realizamos la modificación de nuestro archivo “build.gradle” agregando en la sección de “dependencias” la siguiente línea:

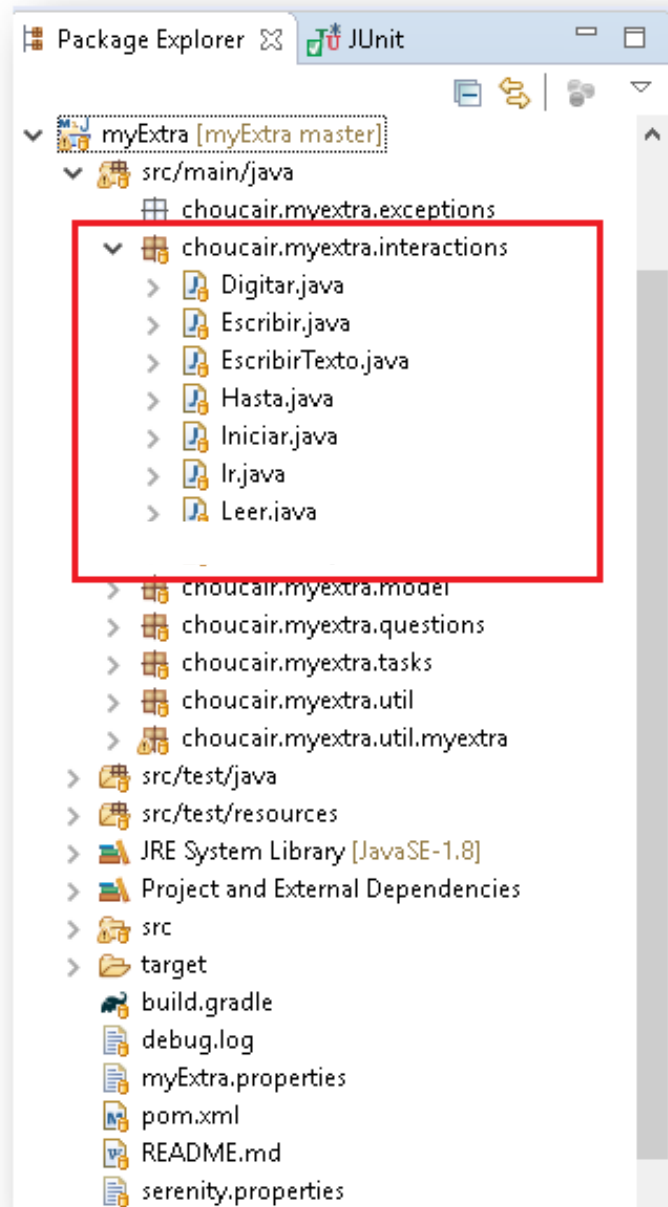
compile group: 'org.jvnet.com4j', name: 'com4j', version: '2.1'

```
45 dependencies {
46     testCompile 'net.serenity-bdd:serenity-core:1.1.1'
47     testCompile 'net.serenity-bdd:serenity-junit:1.1.1'
48     testCompile('junit:junit:4.12')
49     testCompile('org.assertj:assertj-core:1.7.0')
50     testCompile('org.slf4j:slf4j-simple:1.7.7')
51
52     compile 'net.serenity-bdd:serenity-core:1.9.9'
53     compile 'net.serenity-bdd:serenity-junit:1.9.9'
54     compile 'net.serenity-bdd:serenity-cucumber:1.9.5'
55     compile 'net.serenity-bdd:serenity-screenplay:1.9.9'
56     compile 'net.serenity-bdd:serenity-screenplay-webdriver:1.9.9'
57     compile group: 'org.jvnet.com4j', name: 'com4j', version: '2.1'
58 }
59 }
```



Para trabajar con Myextra agregaremos las siguientes clases en el paquete **"interactions"**:

- Digitar
- Escribir
- EscribirTexto
- Hasta
- Iniciar
- Ir
- Leer



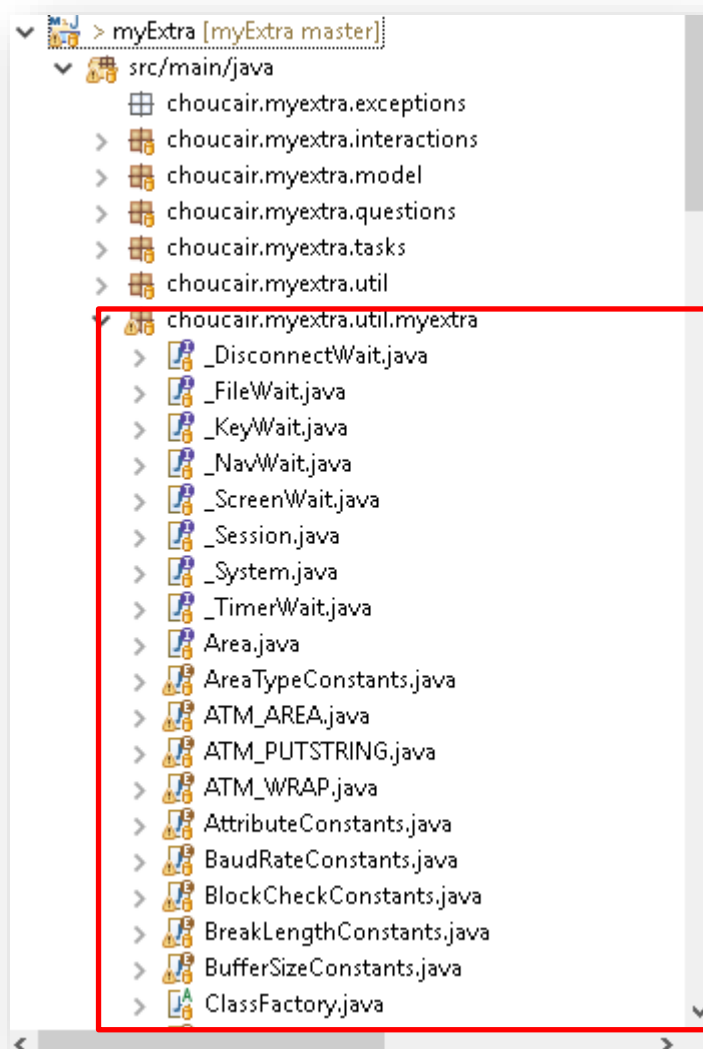
Y la clase **Evidencia**, que la incluiremos en el paquete “**util**”.

Lo anterior ya está montado en un proyecto base que lo puedes descargar del siguiente link y cargarlo como un proyecto GRADLE en el IDE Eclipse.

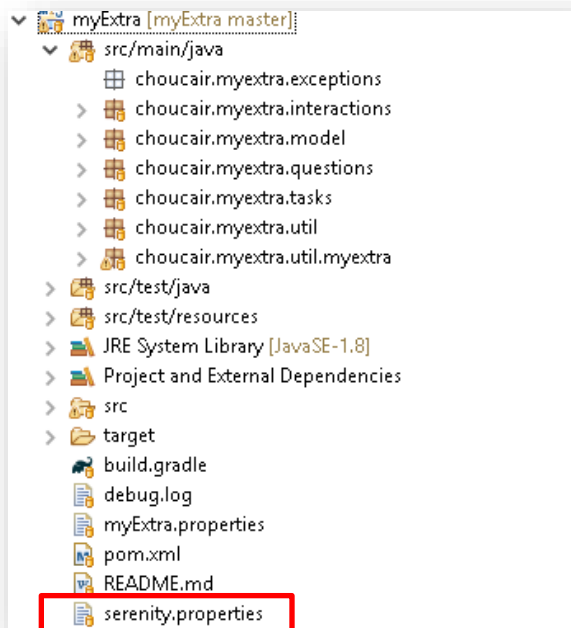
Nota: Recuerda que el proceso de carga se explica en la Guía 1 de Screenplay y debes estar conectado a una **Red de libre acceso a Internet**.

https://drive.google.com/open?id=13D-eVx-K47OHRtpZacCBM_xLd1qyT2pj

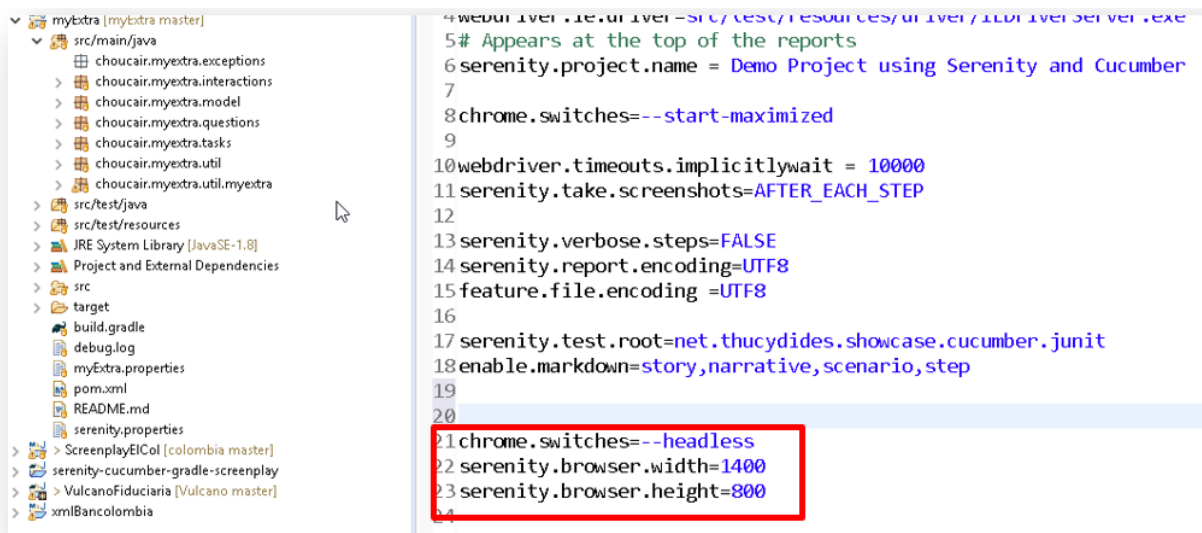
El paquete “**myextra**” quedara incluido dentro del paquete “**util**”.



El archivo “serenity.properties” será modificado



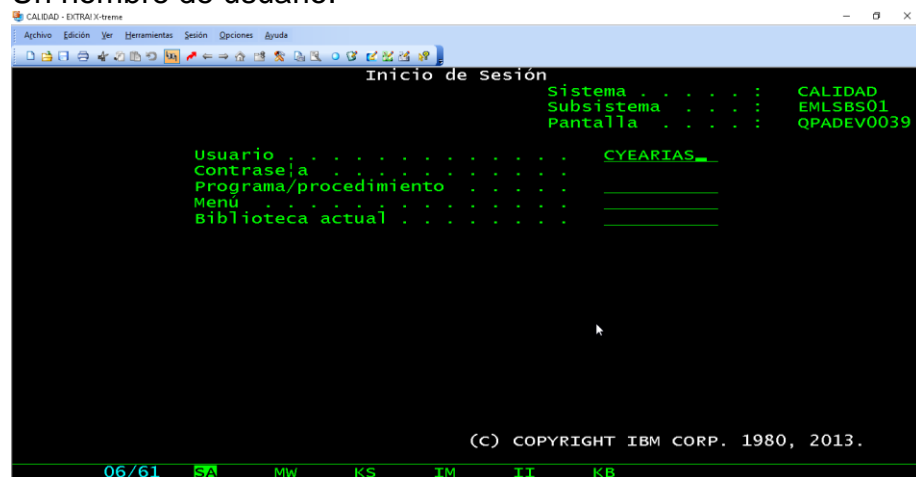
Agregándole las siguientes líneas:



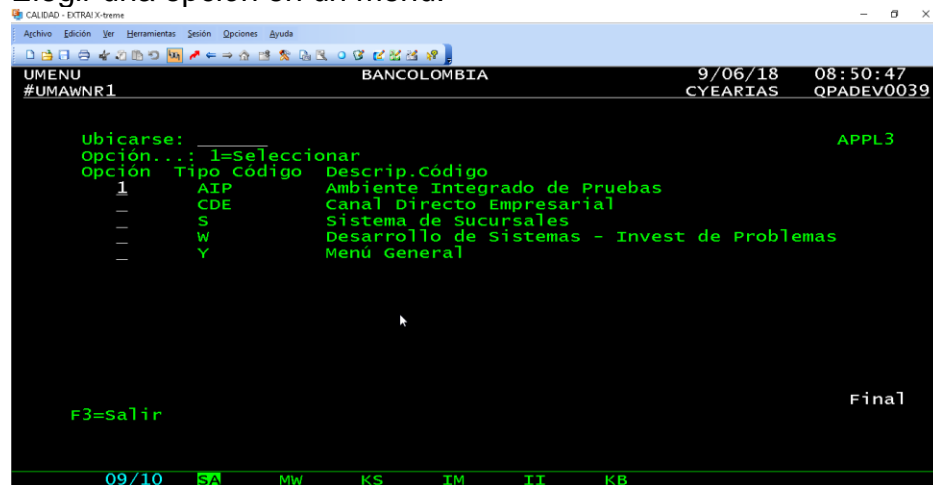
Para trabajar en myextra existen 3 sentencias básicas que nos ayudan a interactuar con cada pantalla. Estas son:

1. LA CAPACIDAD DE ESCRIBIR UN TEXTO.

Un nombre de usuario.



Elegir una opción en un menú.



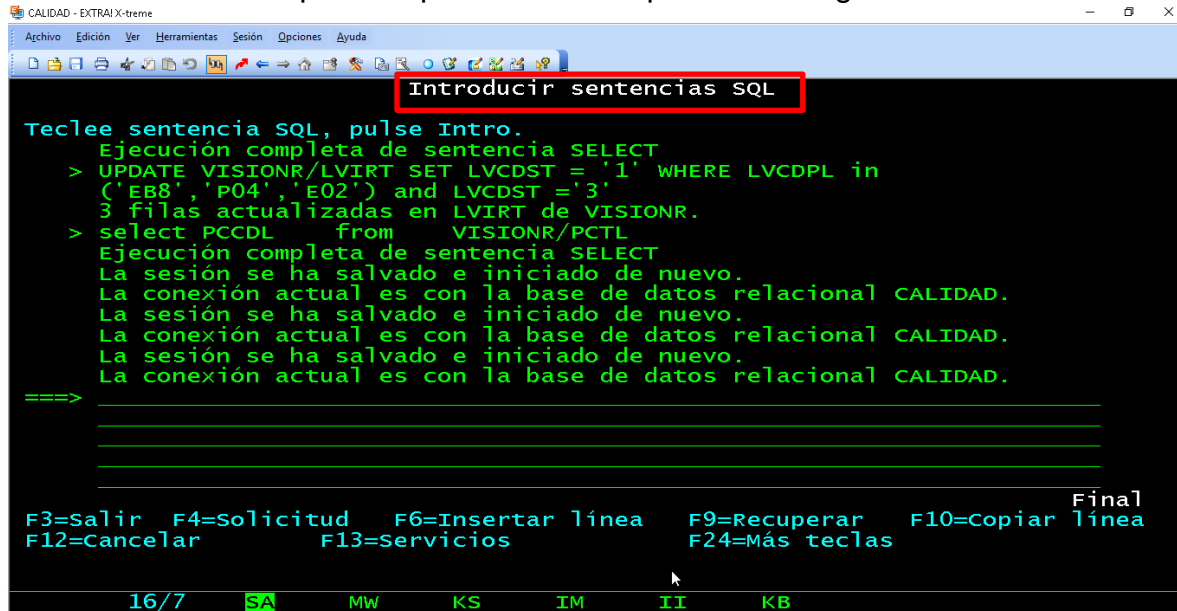
2. PRESIONAR UN TECLA

Generalmente en cada pantalla, luego de escribir un texto debemos confirmar la decisión presionando la tecla "Enter", en muchas otras ocasiones, requerimos salir de la sesión y debemos presionar "F3", o tal vez buscar un registro después de hacer una consulta y presionamos "Av. Pag".



3. LEER TEXTO

Leer el texto de una pantalla para confirmar que hemos llegado al sitio correcto.



The screenshot shows a terminal window with a blue title bar and a black background. The title bar contains the text 'CALIDAD - EXTRAJ X-treme' and standard window controls. The terminal content is as follows:

```
Introducir sentencias SQL

Teclee sentencia SQL, pulse Intro.
Ejecución completa de sentencia SELECT
> UPDATE VISIONR/LVIRT SET LVCDST = '1' WHERE LVCDPL in
('EB8','P04','E02') and LVCDST='3'
3 filas actualizadas en LVIRT de VISIONR.
> select PCCDL from VISIONR/PCTL
Ejecución completa de sentencia SELECT
La sesión se ha salvado e iniciado de nuevo.
La conexión actual es con la base de datos relacional CALIDAD.
La sesión se ha salvado e iniciado de nuevo.
La conexión actual es con la base de datos relacional CALIDAD.
La sesión se ha salvado e iniciado de nuevo.
La conexión actual es con la base de datos relacional CALIDAD.

===>
_____
_____
_____
_____

F3=Salir F4=Solicitud F6=Insertar línea F9=Recuperar F10=Copiar línea Final
F12=Cancelar F13=Servicios F24=Más teclas

16/7 SA MW KS IM II KB
```

**¡Entonces veamos cómo trabajaremos a nivel de código con
screenplay Myextra!**



1. Crearemos una historia de usuario.

```
1 # language:es
2 #Author: your.email@your.domain.com
3
4 Característica: Aprendiendo a usar my extra
5
6 @CasoExitoso
7 Escenario: Ingresar a My Extra
8 Dado que Rafa desea realizar consultas en AS400, Rafa abre my extra
9 Cuando él ingresa sus credenciales
10 Y Navega en un menu
11 Entonces verifica que inició sesión correctamente
12
```

2. Creamos las instrucciones en nuestros StepDefinition

NOTA: La palabra “Iniciar” no puede ser usada para ninguna tarea, pues es una palabra reservada por esta solución para abrir Myextra.

```
21 public class MyExtraStepDefinition {
22
23
24 @Managed(driver="chrome")
25 private WebDriver suNavegador;
26 private Actor rafa= Actor.named("Rafa");
27
28
29 @Before public void setup()
30 {
31     rafa.can(BrowseTheWeb.with(suNavegador));
32 }
33
34 @Dado("^que Rafa desea realizar consultas en AS400, Rafa abre my extra$")
35 public void que_Rafa_desea_realizar_consultas_en_AS_Rafa_abre_my_extra(){
36     rafa.attemptsTo(Abrir.myExtra());
37 }
38
39
40 @Cuando("^él ingresa sus / *$")
```

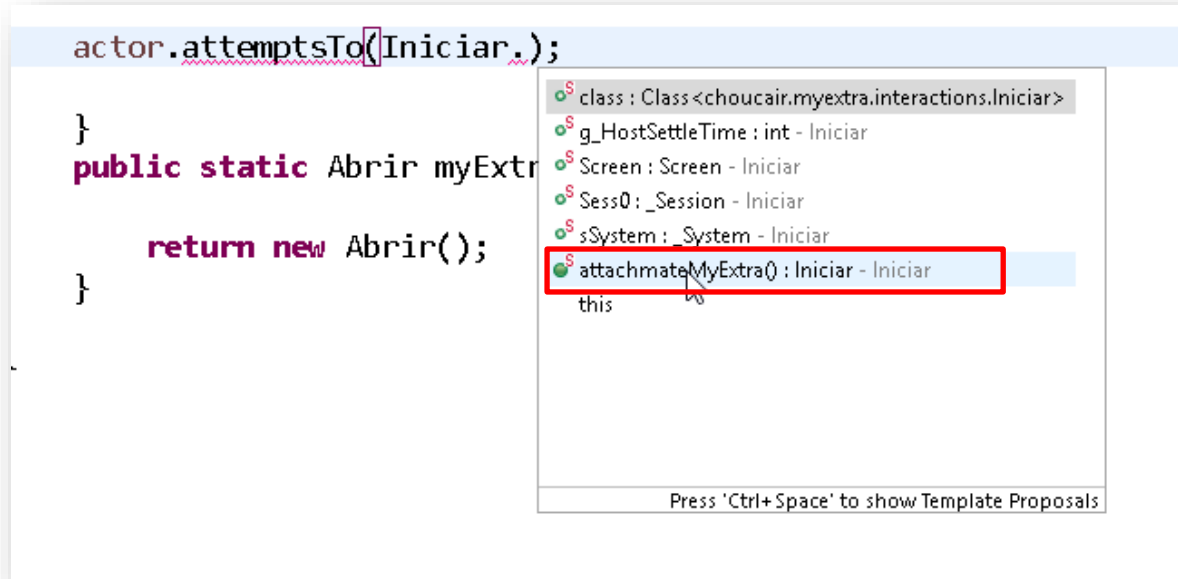
1. Estas líneas son necesarias, pues nos ayudarán con la generación de los reportes para las evidencias.
2. Copiamos los métodos sugeridos a través de la ejecución de la feature y para este primero indicamos la instrucción para Abrir Myextra teniendo en cuenta la legibilidad de la instrucción.



3. Implementamos la tarea “Abrir” y hacemos uso de la interaction “Iniciar”.

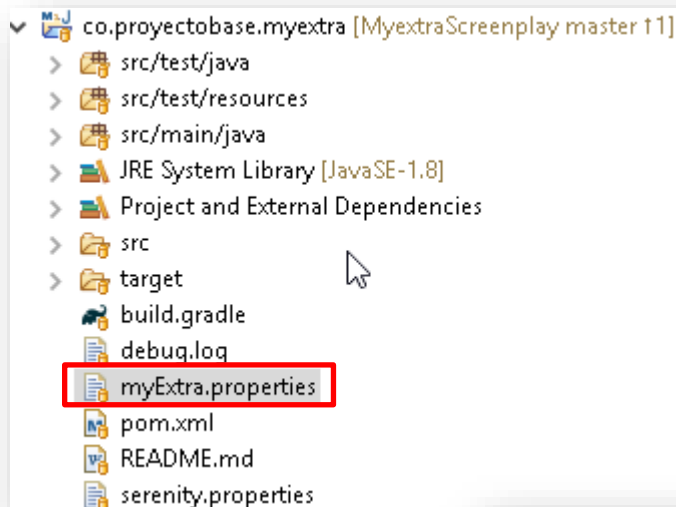
```
7 public class Abrir implements Task{
8
9     @Override
10    public <T extends Actor> void performAs(T actor) {
11
12        actor.attemptsTo(Iniciar.attachmateMyExtra());
13
14    }
15    public static Abrir myExtra() {
16
17        return new Abrir();
18    }
19
20 }
21
```

La interaction “Iniciar” nos ayudará a abrir el MyExtra a través de su método attachmateMyExtra();




```
7 public class Abrir implements Task{
8
9     @Override
10    public <T extends Actor> void performAs(T actor) {
11
12        actor.attemptsTo(Iniciar.attachmateMyExtra());
13
14    }
15    public static Abrir myExtra() {
16
17        return new Abrir();
18    }
19 }
20 }
21 }
```

Nota: Debemos crear un archivo llamado “myExtra.properties” Dentro de la raíz del proyecto. ¡Y dentro del mismo creamos un atributo llamado “rutaCalidad” donde definiremos la ruta desde donde se abrirá nuestra sesión de MyExtra! Y otro atributo llamado “**rutaEvidencia**”, donde definiremos la ruta que nos servirá de apoyo para la generación del reporte de Serenity (La Ruta que allí coloquemos deberá estar creada físicamente en nuestro PC).



```
1 rutaCalidad=C://Users//rperezr//Desktop//CALIDAD.EDP
2 rutaEvidencia=D://evidencia//evidencia.png
```

4. Seguimos con las demás instrucciones en nuestro StepDefinition

```
public class MyExtraStepDefinition {  
  
    @Managed(driver="chrome")  
    private WebDriver suNavegador;  
    private Actor rafa= Actor.named("Rafa");  
  
    @Before public void setup()  
    {  
        rafa.can(BrowseTheWeb.with(suNavegador));  
    }  
  
    @Dado("^que Rafa desea realizar consultas en AS400, Rafa abre my extra$")  
    public void que_Rafa_desea_realizar_consultas_en_AS_Rafa_abre_my_extra(){  
        rafa.attemptsTo(Abrir.myExtra());  
    }  
  
    @Cuando("^él ingresa sus credenciales$")  
    public void él_ingresa_sus_credenciales() {  
        rafa.attemptsTo(Loguearse.enMyExtra());  
    }  
}
```



5. Creamos e Implementamos la tarea “**Loguearse**” en nuestro paquete “**Tasks**”

```
public class Loguearse implements Task{

    private List<Credenciales> datos;

    public Loguearse(List<Credenciales> datos) {
        this.datos = datos;
    }

    @Override
    public <T extends Actor> void performAs(T actor) {
        actor.attemptsTo(Escribir.elTexto(datos.get(0).getusuario()).en(USUARIO));
        actor.attemptsTo(Escribir.elTexto(datos.get(0).getcontraseña()).en(PASSWORD));
        actor.attemptsTo(Digitar.laTecla(Tecla.ENTER.getTecla()));
        actor.attemptsTo(Ir.hastaVerTexto(datos.get(0).getTexto()).en(UBICACION));
    }

    public static Loguearse enMyExtra(List<Credenciales> datos) {
        return Tasks.instrumented(Loguearse.class, datos);
    }
}
```

Y para escribir en pantalla usaremos la interaction “**Escribir**” con la estructura mostrada anteriormente. Primero le indicaremos el texto que queremos escribir.

```
public class Loguearse implements Task{

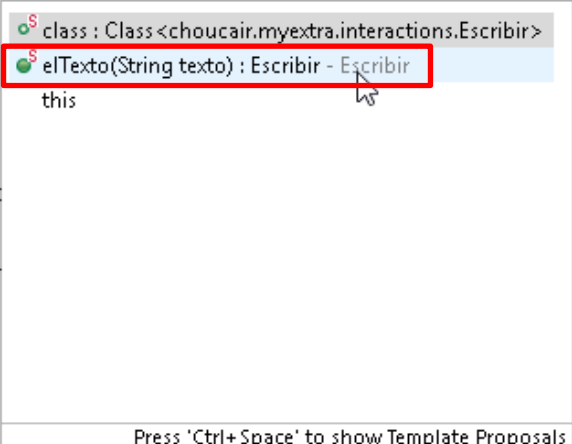
    @Override
    public <T extends Actor> void performAs(T actor) {

        actor.attemptsTo(Escribir.);

    }

    public static Loguearse enMyExtra(List<Credenciales> datos) {

        return Tasks.instrumented(Loguearse.class, datos);
    }
}
```



Press 'Ctrl+Space' to show Template Proposals

Y luego le diremos dónde tiene que escribirlo.

```
@Override
public <T extends Actor> void performAs(T actor) {
    actor.attemptsTo(Escribir.elTexto("XXXXXXXXXXXX"));
    actor.attemptsTo(Escribir.elTexto("*****"));
    actor.attemptsTo(Digitar.laTecla(Tecla.ENTER.toString()));
    actor.attemptsTo(Ir.hastaVerTexto(datos.get(0).getTexto()));
}

public static Loguearse enMyExtra(List<Credenciales> datos) {
    return Tasks.instrumented(Loguearse.class, datos);
}
```

en(Target target) : EscribirTexto - Escribir
equals(Object obj) : boolean - Object
getClass() : Class<?> - Object
hashCode() : int - Object
toString() : String - Object
notify() : void - Object
notifyAll() : void - Object
wait() : void - Object
wait(long timeout) : void - Object
wait(long timeout, int nanos) : void - Object
elTexto(String texto) : Escribir - Escribir
new

Press 'Ctrl+Space' to show Template Proposals

```
public class Loguearse implements Task{

    private List<Credenciales> datos;

    public Loguearse(List<Credenciales> datos) {
        this.datos = datos;
    }

    @Override
    public <T extends Actor> void performAs(T actor) {
        actor.attemptsTo(Escribir.elTexto("XXXXXXXXXXXX").en(USUARIO));
        actor.attemptsTo(Escribir.elTexto("*****").en(PASSWORD));
        actor.attemptsTo(Digitar.laTecla(Tecla.ENTER.getTecla()));
        actor.attemptsTo(Ir.hastaVerTexto(datos.get(0).getTexto()).en(UBICACION));
    }

    public static Loguearse enMyExtra(List<Credenciales> datos) {
        return Tasks.instrumented(Loguearse.class, datos);
    }
}
```



Como vemos, el texto lo escribiremos en un Target, dicho Target debe ser mapeado en una clase "Page" que creamos en nuestro paquete "**userinterface**". Usamos el método "locatedBy" e indicamos las coordenadas (row y col).

```
import co.proyectobase.myextra.model.Target;

public class LoginPage {

    public static final Target USUARIO =Target.the("usuario").locatedBy(6, 53);
    public static final Target PASSWORD =Target.the("contraseña").locatedBy(7, 53);
    public static final Target UBICACION=Target.the("ubicarse").locatedBy(5, 7);
}
```

Cabe resaltar que esta clase Target es una clase personalizada, es decir, no es la misma Target que usamos normalmente por el Screenplay en web.

También tenemos dos interactions ya implementadas que nos ayudarán a Digitar y navegar entre pantallas hasta que aparezca un determinado texto en determinada posición.

Estos métodos podremos usarlos en cualquier tarea que deseemos enviándole la tecla que deseamos presionar, y pasándole el texto que queremos ver en pantalla. Dentro del paquete "**model**" encontraremos una clase llamada "Tecla", en dicha clase agregaremos cada tecla que necesitemos para nuestra automatización. Para el ejemplo dejaremos agregadas las teclas "ENTER" y "F12".

```
public class Loguearse implements Task{

    private List<Credenciales> datos;

    public Loguearse(List<Credenciales> datos) {
        this.datos = datos;
    }

    @Override
    public <T extends Actor> void performAs(T actor) {
        actor.attemptsTo(Escribir.elTexto("XXXXXXXXXX").en(USUARIO));
        actor.attemptsTo(Escribir.elTexto("*****").en(PASSWORD));
        actor.attemptsTo(Digitar.laTecla(Tecla.ENTER.getTecla()));
        actor.attemptsTo(Ir.hastaVerTexto(datos.get(0).getTexto()).en(UBICACION));
    }

    public static Loguearse enMyExtra(List<Credenciales> datos) {
        return Tasks.instrumented(Loguearse.class, datos);
    }
}
```

```
public enum Tecla {  
    ENTER ("ENTER"),  
    F12 ("Pf12");  
  
    private final String tecla;  
  
    Tecla(String tecla)  
    {  
        this.tecla=tecla;  
    }  
  
    public String getTecla()  
    {  
        return tecla;  
    }  
}
```

Como podemos ver la tecla ENTER la hemos asignado como un String "ENTER", sin embargo, el F12 no le corresponde su mismo nombre en el String, la forma en que MyExtra! Lo identifica es "Pf12". Para saber cómo declarar otras teclas puedes visitar la siguiente página.

http://docs.attachmate.com/extra/x-treme/apis/com/5250functionkeys_des.htm



Veamos cómo navegar entre menús.

6. Creamos e implementamos una tarea llamada "Navegar"

Escribimos en nuestra stepDefinition la siguiente instrucción:

```
public class MyExtraStepDefinition {

    @Managed(driver="chrome")
    private WebDriver suNavegador;
    private Actor rafa= Actor.named("Rafa");

    @Before public void setup()
    {
        rafa.can(BrowseTheWeb.with(suNavegador));
    }

    @Dado("^que Rafa desea realizar consultas en AS400, Rafa abre my extra$")
    public void que_Rafa_desea_realizar_consultas_en_AS_Rafa_abre_my_extra(){
        rafa.attemptsTo(Abrir.myExtra());
    }

    @Cuando("^él ingresa sus credenciales$")
    public void él_ingresa_sus_credenciales() {

        rafa.attemptsTo(Loguearse.enMyExtra());

    }

    @Cuando("^Navega en un menu$")
    public void navega_en_un_Menu()
    {
        rafa.attemptsTo(Navegar.aUnMenu());
    }
}
```



Y en nuestros Task “Navegar” al igual que en loguearse, hacemos uso de Escribir, Digitar e Ir.

```
public class Navegar implements Task{

    @Override
    public <T extends Actor> void performAs(T actor) {
        actor.attemptsTo(Escribir.elTexto("1").en(AMBIENTE_PRUEBAS));
        actor.attemptsTo(Digitar.laTecla(Tecla.ENTER.getTecla()));
        actor.attemptsTo(Escribir.elTexto("1").en(CONFIRMACION_AMBIENTE_PRUEBAS));
        actor.attemptsTo(Digitar.laTecla(Tecla.ENTER.getTecla()));
        actor.attemptsTo(Escribir.elTexto("1").en(SQL));
        actor.attemptsTo(Digitar.laTecla(Tecla.ENTER.getTecla()));
        actor.attemptsTo(Ir.hastaVerTexto("SQL").en(TITULO_SQL_PAGE));
    }

    public static Navegar aUnMenu() {
        return Tasks.instrumented(Navegar.class);
    }
}
```

Cabe resaltar que debemos crear nuestros Targets “AMBIENTE_PRUEBAS”, “CONFIRMACIÓN_AMBIENTE_PRUEBAS”, “SQL” y “TITULO_SQL_PAGE” en una clase Page dentro del paquete “userinterface”.

```
public class MenuPage {
    public static final Target AMBIENTE_PRUEBAS = Target.the("Ambiente integrado de pruebas").locatedBy(8, 10);
    public static final Target CONFIRMACION_AMBIENTE_PRUEBAS = Target.the("Confirmación").locatedBy(18, 8);
    public static final Target SQL = Target.the("Ambiente SQL").locatedBy(13, 9);
}
```

```
public class SQLPage {
    public static final Target TITULO_SQL_PAGE = Target.the("Ambiente SQL").locatedBy(1, 50);
}
```



Por último haremos una instrucción para verificar, crearemos una Question y usaremos una interaction que nos traiga un texto.

```
public class MyExtraStepDefinition {

    @Managed(driver="chrome")
    private WebDriver suNavegador;
    private Actor rafa= Actor.named("Rafa");
    @Before public void setup()
    {
        rafa.can(BrowseTheWeb.with(suNavegador));
    }

    @Dado("^que Rafa desea realizar consultas en AS400, Rafa abre my extra$")
    public void que_Rafa_desea_realizar_consultas_en_AS_Rafa_abre_my_extra(){
        rafa.attemptsTo(Abrir.myExtra());
    }

    @Cuando("^él ingresa sus credenciales$")
    public void él_ingresa_sus_credenciales() {
        rafa.attemptsTo(Loguearse.enMyExtra());
    }

    @Cuando("^Navega en un menu$")
    public void navega_en_un_Menu()
    {
        rafa.attemptsTo(Navegar.aUnMenu());
    }

    @Entonces("^verifica que inició sesión correctamente$")
    public void verifica_que_inició_sesión_correctamente(){
        rafa.should(seeThat(LaRespuesta.es(), equalTo("Introducir sentencias SQL")));
    }
}
```



En nuestra clase LaRespuesta usaremos la clase Leer y su método “elTextoEn” enviando como parámetro un Target que debe ser mapeado en un Page dentro de nuestro paquete userinterface:

```
public class LaRespuesta implements Question<String>{  
  
    @Override  
    public String answeredBy(Actor actor) {  
        return Leer.elTextoEn(SQLPage.TITULO);  
    }  
  
    public static LaRespuesta es() {  
        return new LaRespuesta();  
    }  
}
```

Para leer Targets tendremos que enviar una longitud, entonces al momento de mapear nuestra Target usaremos el método “**locatedBy**” en su sobrecarga que contiene el parámetro “**length**”.

```
public static final Target TITULO=Target.the("SQL").located  
    locatedBy(int row, int col) : Target - Target  
    locatedBy(int row, int col, int length) : Target - Target
```

Press 'Ctrl+Space' to show Template Proposals

Con estas interacciones básicas son muchas las tareas que puedes implementar, y serán de gran utilidad para tus proyectos de automatización en MyExtra!

¡Adelante...ponlas en práctica!

