

GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

Clonación del proyecto base con patron de diseño POM y configuración del proyecto

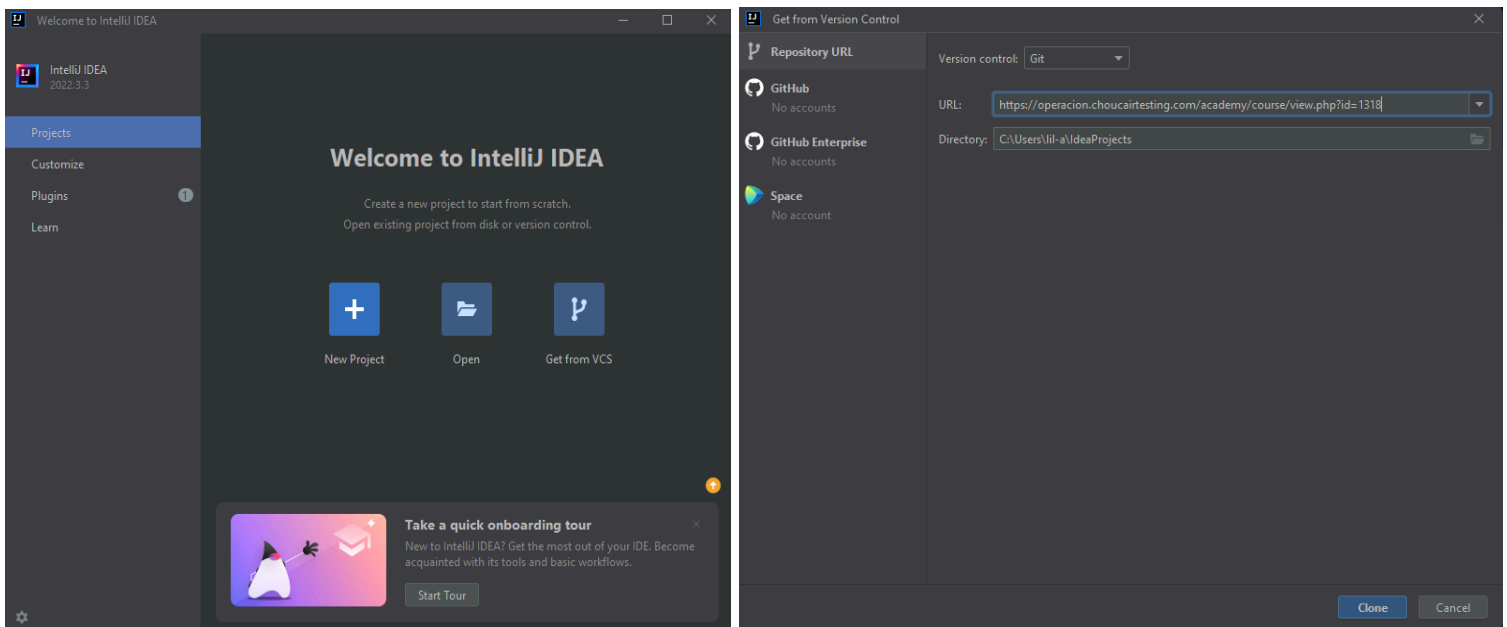
Antes de iniciar tenemos que saber que es Serenity BDD, Serenity es una biblioteca de código abierto que le ayuda a redactar pruebas de aceptación automatizadas de mayor calidad de forma más rápida. Serenity te ayuda a:

- Escribir pruebas que sean más flexibles y fáciles de mantener
- Producir informes ilustrados y narrativos sobre sus pruebas
- Asigne sus pruebas automatizadas a sus requisitos
- Vea cuánto de su aplicación se está probando realmente
- Y controle el progreso del proyecto

Ahora si iniciamos abriendo el IDE IntelliJ

Paso # 1:

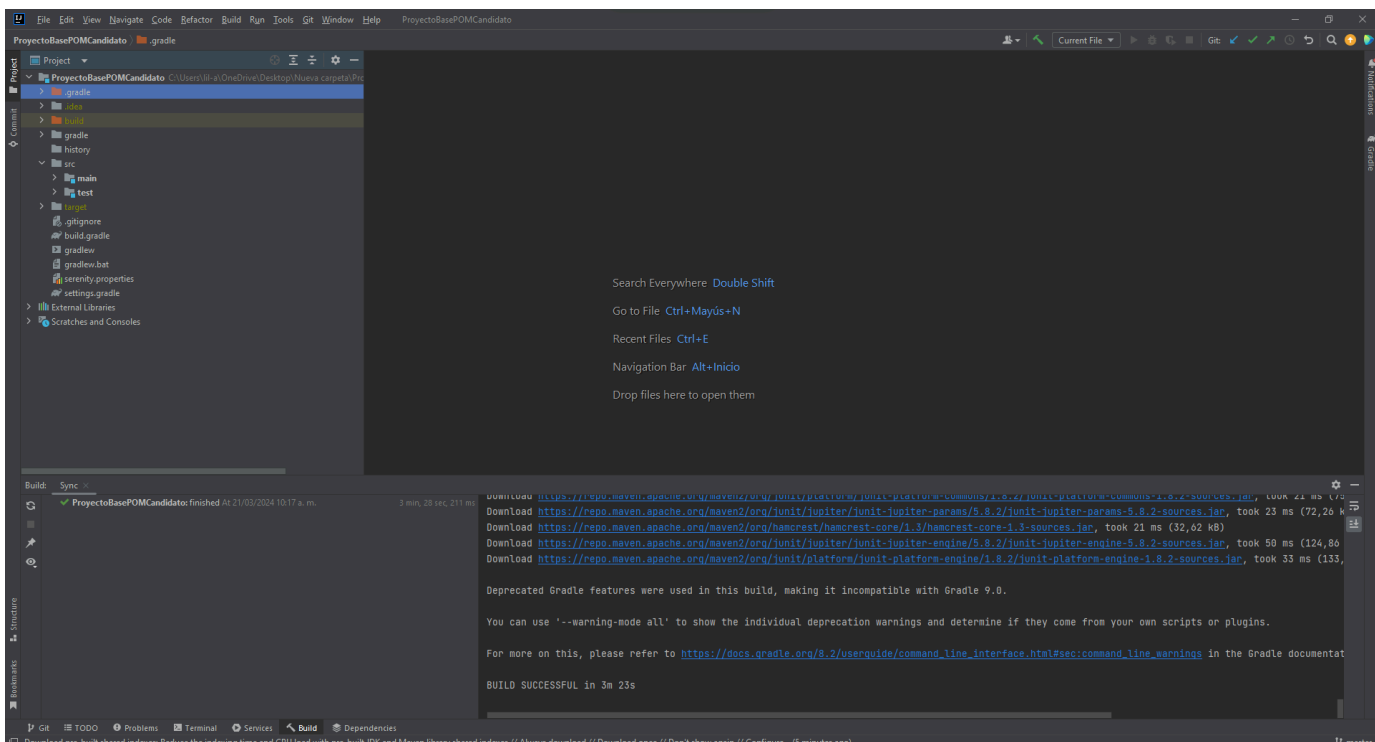
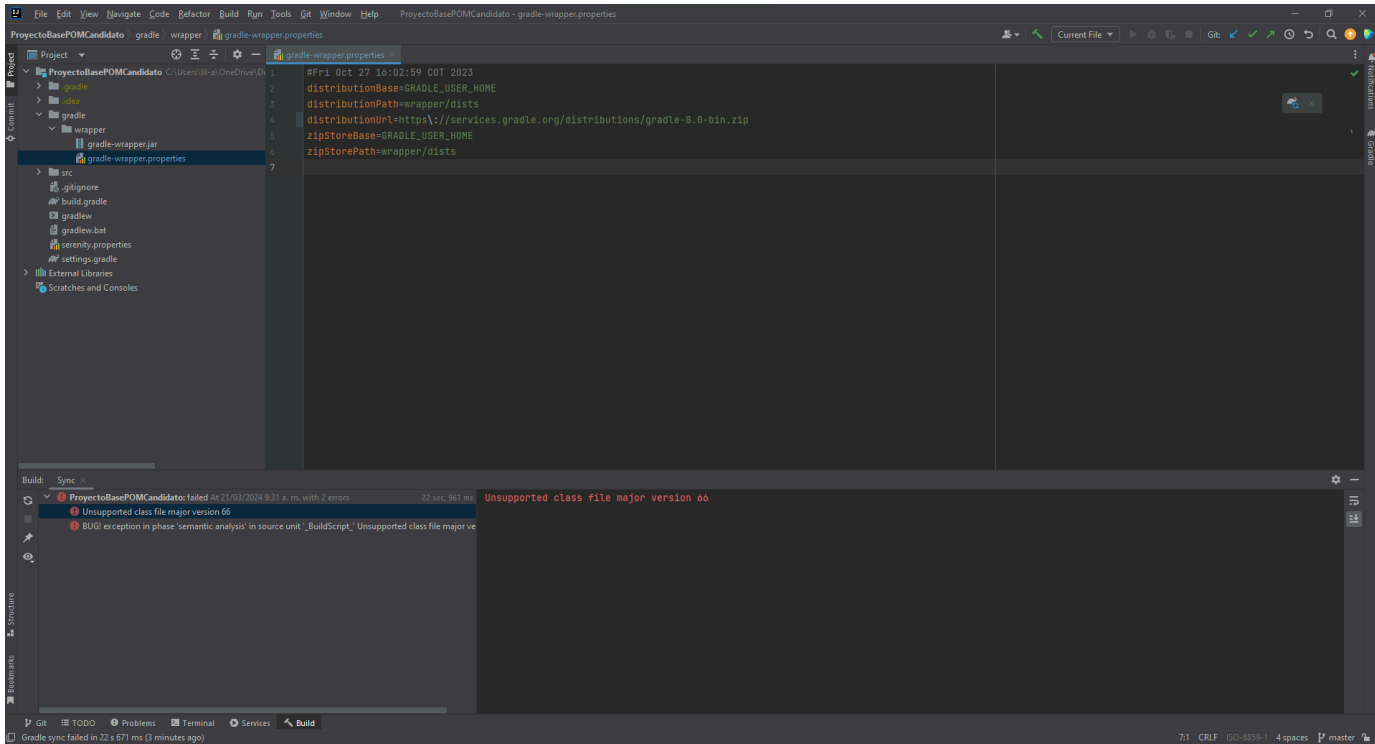
Debemos **clonar** el repositorio que encontramos en la sección de **Requisitos** directamente en la carpeta donde deseamos almacenar el proyecto o a través del entorno de desarrollo, al abrir IntelliJ podemos clonar el repositorio desde la opción **Get from VCS**. acá pueden suceder dos cosas si GIT está instalado te va a permitir poner la URL del proyecto, si no está instalado puedes instalarlo en ese momento. Una vez lo tengan instalado pueden dar click en el botón Clone y esperar a que el proyecto se compile.



GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

Paso # 2 :

Una vez Finalizado la creación del proyecto, esperamos ya que este empezara a cargar la configuración base para proyectos **Gradle** y a su vez iniciara con la compilación de esa misma configuración por defecto que crea el IDE. si te encuentras con el siguiente error: “Unsupported class file major version ##” es porque la version de Gradle del proyecto está mal configurada. para esto nos dirigimos a la carpeta **gradle** que tenemos en la barra izquierda y abrimos el archivo **gradle-wrapper.properties** y cambiamos la version de gradle a 8.0 en la línea **DistributionUrl** y refrescamos el proyecto.



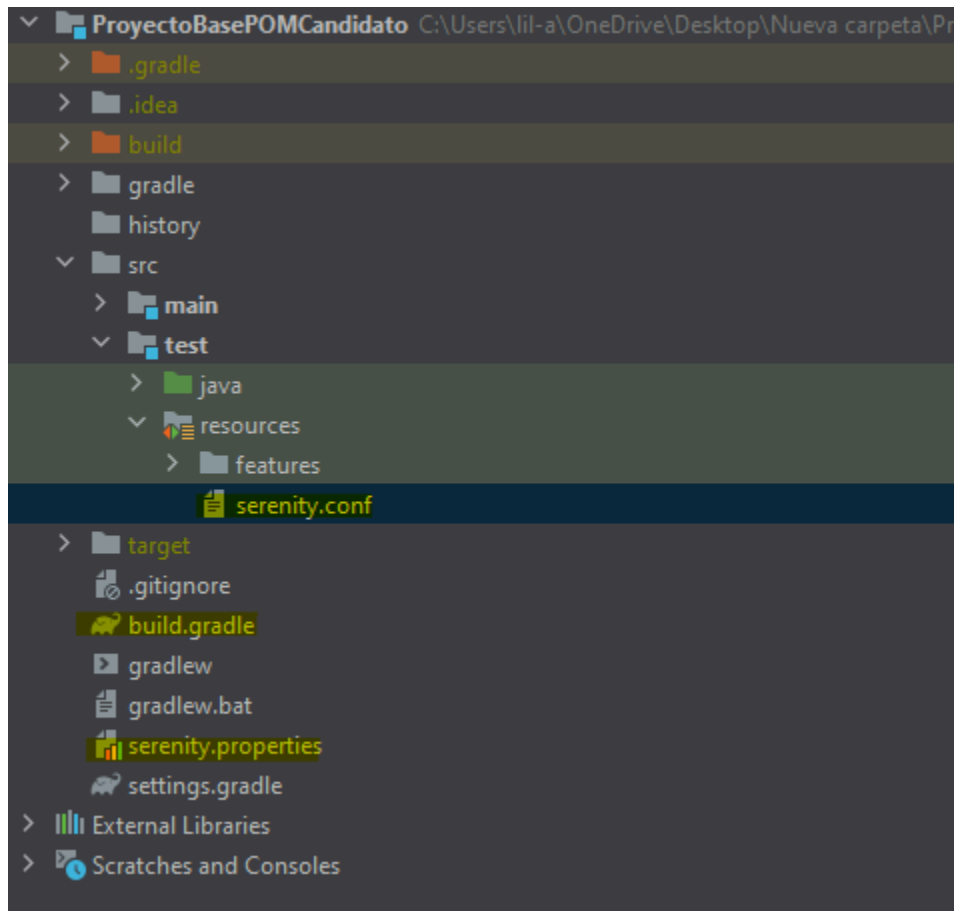
GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

El proyecto contiene 3 archivos que debemos tener en cuenta para la configuración. los archivos son nombrados de la siguiente manera:

serenity.properties -> contiene propiedades de ajustes para nuestro proyecto

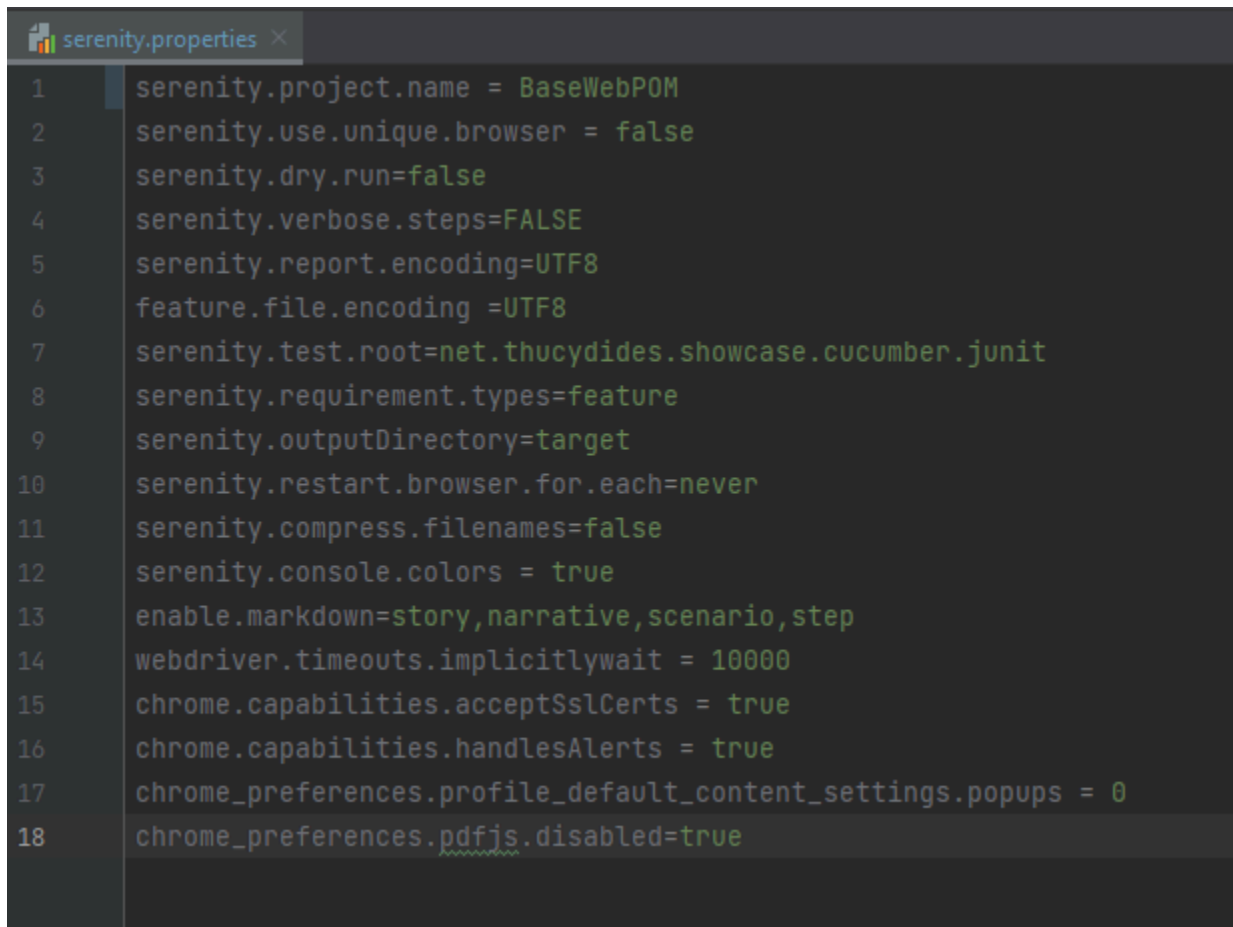
serenity.conf -> contiene propiedades de navegador y variables de ambiente.

build.gradle -> en este tenemos la configuración y dependencias de nuestro proyecto.



GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

- serenity.properties



```
1 serenity.project.name = BaseWebPOM
2 serenity.use.unique.browser = false
3 serenity.dry.run=false
4 serenity.verbose.steps=FALSE
5 serenity.report.encoding=UTF8
6 feature.file.encoding =UTF8
7 serenity.test.root=net.thucydides.showcase.cucumber.junit
8 serenity.requirement.types=feature
9 serenity.outputDirectory=target
10 serenity.restart.browser.for.each=never
11 serenity.compress_filenames=false
12 serenity.console.colors = true
13 enable.markdown=story,narrative,scenario,step
14 webdriver.timeouts.implicitlywait = 10000
15 chrome.capabilities.acceptSslCerts = true
16 chrome.capabilities.handlesAlerts = true
17 chrome_preferences.profile_default_content_settings.popups = 0
18 chrome_preferences.pdfjs.disabled=true
```

NOTA: A continuación se definen cada una de las opciones dadas en el archivo serenity.properties

GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

Definicion de cada linea del `serenity.properties`

`serenity.project.name`: Nombre del proyecto en Serenity.

`serenity.use.unique.browser = false`: No se utiliza un navegador único para todas las pruebas; esto significa que las pruebas no comparten la misma sesión del navegador.

`serenity.dry.run=false`: No se ejecuta una simulación (prueba seca) donde no se realizan cambios reales.

`serenity.verbose.steps=FALSE`: No se muestran detalles exhaustivos de los pasos de las pruebas

`serenity.report.encoding=UTF8`: La codificación utilizada para los informes de Serenity

`feature.file.encoding =UTF8`: La codificación utilizada para los archivos `.feature`

`serenity.test.root=net.thucydides.showcase.cucumber.junit`: Directorio raíz de las pruebas para un proyecto de muestra de Cucumber y JUnit.

`serenity.requirement.types=feature`: Tipos de requisitos utilizados en Serenity.

`serenity.outputDirectory=target`: Directorio de salida para los informes de Serenity

`serenity.restart.browser.for.each=never`: El navegador no se reinicia para cada prueba.

`serenity.compress.filesnames=false`: Los nombres de archivo no se comprimen.

`serenity.console.colors = true`: Se utilizan colores en la consola de Serenity.

`enable.markdown=story,narrative,scenario,step`: Markdown está habilitado

`webdriver.timeouts.implicitlywait = 10000`: Tiempo de espera implícito para WebDriver es de 10000 milisegundos (10 segundos).

`chrome.capabilities.acceptSslCerts = true`: Chrome acepta certificados SSL.

`chrome.capabilities.handlesAlerts = true`: Chrome maneja alertas.

`chrome_preferences.profile_default_content_settings.popups = 0`: Configuración predeterminada de Chrome para la gestión de ventanas emergentes es 0.

`chrome_preferences.pdfjs.disabled=true`: PDF.js está deshabilitado en Chrome.

GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

- serenity.conf

```
serenity.conf x
1  webdriver {
2      driver = chrome
3      autodownload = true
4  }
5
6  headless.mode = false
7  logging.level.org.opengauss.selenium= INFO
8  chrome_preferences.profile.default_content_setting_values.geolocation = 1
9  logging.level.net.thucydides = INFO
10
11 serenity {
12     encoding = "UTF-8"
13     compress_filenames = true
14     take_screenshots = FOR_EACH_ACTION
15 }
16
17 webdriver {
18     capabilities {
19         "goog:chromeOptions" {
20             args = [ "start-maximized", "test-type", "no-sandbox", "disable-popup-blocking",
21                 "disable-download-notification", "ignore-certificate-errors", "allow-running-insecure-content", "disable-translate",
22                 "always-authorize-plugins", "disable-extensions", "disable-notifications", "disable-geolocation", "enable-strict-powerful-feature-restrictions"
23             ]
24         }
25     }
26 }
27 environments {
28     qa {
29         base.url = "https://demo.serenity.is/Account/Login?ReturnUrl=%2F"
30         base.user = user
31         base.pass = pass
32     }
33 }
```

NOTA: A continuación se definen cada una de las opciones dadas en el archivo serenity.conf

GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

```
webdriver {  
  driver=chrome  
  autodownload = true  
}
```

Esta sección le dice a WebDriver que use el navegador Chrome para las pruebas y que descargue automáticamente los controladores de Chrome si no están presentes.

headless.mode = false: Esta línea indica que las pruebas se deben ejecutar en modo no "headless" (es decir, las pruebas se ejecutan abriendo el navegador). Si se configura como verdadero, las pruebas se ejecutarán en segundo plano sin abrir el navegador.

logging.level.org.openqa.selenium= INFO: Configura el nivel de registro para Selenium. "INFO" es un nivel de registro que proporciona información sobre lo que está sucediendo durante la ejecución.

chrome_preferences.profile.default_content_setting_values.geolocation = 1: Esto permite la geolocalización en Chrome.

logging.level.net.thucydides = INFO: Configura el nivel de registro para Thucydides (una antigua denominación de Serenity BDD). "INFO" es un nivel de registro que proporciona información sobre lo que está sucediendo durante la ejecución.

```
serenity {  
  encoding =  
  "UTF-8"  
  compress_filenames = true  
  take_screenshots =  
  FOR_EACH_ACTION  
}
```

Esta sección configura Serenity para usar la codificación UTF-8, comprimir los nombres de archivo y tomar capturas de pantalla en cada acción durante las pruebas.

```
webdriver {  
  capabilities {  
    "goog:chromeOptions"  
    { args = [ ... ]  
    }  
  }  
}
```

Esta sección configura varias opciones para Chrome. Las opciones incluyen iniciar Chrome maximizado, deshabilitar el bloqueo de ventanas emergentes, ignorar errores de certificado, deshabilitar las notificaciones, deshabilitar la geolocalización, etc.

GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

- build.gradle

```
build.gradle (ProyectoBasePOMCandidato) x
1  group 'BaseWebPOM'
2  version '1.0'
3  apply plugin: 'java-library'
4  apply plugin: 'java'
5  apply plugin: 'groovy'
6  apply plugin: "net.serenity-bdd:serenity-gradle-plugin"
7
8  def versionSerenity :String = '3.9.8'
9  def versionCucumber :String = '3.9.8'
10 compileJava.options.encoding = "UTF-8"
11 compileTestJava.options.encoding = "UTF-8"
12
13 repositories {
14     mavenCentral()
15 }
16
17 buildscript {
18     repositories {
19         maven {
20             url "https://plugins.gradle.org/m2/"
21         }
22     }
23     dependencies {
24         classpath "net.serenity-bdd:serenity-gradle-plugin:3.9.8"
25     }
26 }
27
28 dependencies {
29
30     implementation "net.serenity-bdd:serenity-core:${versionSerenity}"
31     implementation "net.serenity-bdd:serenity-junit:${versionSerenity}"
32     implementation "net.serenity-bdd:serenity-ensure:${versionSerenity}"
33     implementation "net.serenity-bdd:serenity-cucumber:${versionCucumber}"
34     implementation "net.serenity-bdd:serenity-screenplay:${versionSerenity}"
35     implementation "net.serenity-bdd:serenity-report-resources:${versionSerenity}"
36     implementation "net.serenity-bdd:serenity-screenplay-webdriver:${versionSerenity}"
37
38     implementation "io.cucumber:datatable:${versionCucumber}"
39     testImplementation "io.cucumber:cucumber-junit:${versionCucumber}"
40     implementation 'org.junit.jupiter:junit-jupiter:5.8.1'
41     testImplementation 'org.junit.jupiter:junit-jupiter-api:5.8.2'
42     implementation group: 'org.slf4j', name: 'slf4j-api', version: '2.0.0-alpha5'
43     implementation group: 'org.slf4j', name: 'slf4j-simple', version: '2.0.0-alpha5'
44     implementation group: 'org.apache.logging.log4j', name: 'log4j-core', version: '2.18.0'
45     implementation group: 'org.assertj', name: 'assertj-core', version: '3.8.0'
46     implementation group: 'org.hamcrest', name: 'hamcrest-all', version: '1.3'
47     implementation 'io.github.bonigarcia:webdrivermanager:5.5.3'
48     implementation 'com.github.javafaker:javafaker:1.0.2'
49
50 }
51
52 test {
53     useJUnit {
54         include "**/*";
55         gradle.startParameter.continueOnFailure = true
56         testLogging.showStandardStreams = true
57         systemProperties System.getProperties()
58         systemProperty "cucumber.filter.tags", System.getProperty("cucumber.filter.tags")
59     }
60 }
61 }
```

Una vez escrita la configuración, damos click sobre el botón de Gradle para cargar y compilar las nuevas versiones de las librerías implementadas en dicha configuración.

GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

Explicación del Build.Gradle

Group y Version

```
1 group 'BaseWebPOM'
2 version '1.0'
```

Estas líneas definen el grupo y la versión del proyecto, utilizados a menudo para identificar el proyecto en un repositorio.

Plugins

```
3 apply plugin: 'java-library'
4 apply plugin: 'java'
5 apply plugin: 'groovy'
6 apply plugin: "net.serenity-bdd.serenity-gradle-plugin"
```

Estas líneas aplican varios plugins al proyecto.

java: Soporte para proyectos Java.

java-library: Extiende el plugin java y añade soporte para construir bibliotecas Java.

net.serenity-bdd.serenity-gradle-plugin: Soporte para el marco Serenity BDD (Desarrollo Guiado por Comportamiento).

Opciones de Compilación Java y Versiones de dependencias

```
8 def versionSerenity :String ='3.9.8'
9 def versionCucumber :String ='3.9.8'
10 compileJava.options.encoding = "UTF-8"
11 compileTestJava.options.encoding = "UTF-8"
```

Repositorios

```
13 repositories {
14     mavenCentral()
15 }
16
17 buildscript {
18     repositories {
19         maven {
20             url "https://plugins.gradle.org/m2/"
21         }
22     }
23     dependencies {
24         classpath "net.serenity-bdd:serenity-gradle-plugin:3.9.8"
25     }
26 }
```

GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

repositories { mavenCentral() }: Se configura Maven Central como el repositorio de dependencias para el proyecto.

buildscript { ... }: Este bloque define la configuración para el script de construcción de Gradle.

buildscript: se configura un repositorio Maven adicional que apunta a `"https://plugins.gradle.org/m2/"`.

dependencies { ... }: Aquí se definen las dependencias necesarias para la construcción del script.

dependencies, se especifica la dependencia del plugin de Gradle para Serenity BDD, con la versión 3.9.8. Este plugin es utilizado para integrar Serenity BDD con el sistema de construcción de Gradle.

Dependencias

El bloque dependencies enumera todas las dependencias requeridas por el proyecto:

```
28 dependencies {
29
30     implementation "net.serenity-bdd:serenity-core:${versionSerenity}"
31     implementation "net.serenity-bdd:serenity-junit:${versionSerenity}"
32     implementation "net.serenity-bdd:serenity-ensure:${versionSerenity}"
33     implementation "net.serenity-bdd:serenity-cucumber:${versionCucumber}"
34     implementation "net.serenity-bdd:serenity-screenplay:${versionSerenity}"
35     implementation "net.serenity-bdd:serenity-report-resources:${versionSerenity}"
36     implementation "net.serenity-bdd:serenity-screenplay-webdriver:${versionSerenity}"
37
38     implementation "io.cucumber:datatable:${versionCucumber}"
39     testImplementation "io.cucumber:cucumber-junit:${versionCucumber}"
40     implementation 'org.junit.jupiter:junit-jupiter:5.8.1'
41     testImplementation 'org.junit.jupiter:junit-jupiter-api:5.8.2'
42     implementation group: 'org.slf4j', name: 'slf4j-api', version: '2.0.0-alpha5'
43     implementation group: 'org.slf4j', name: 'slf4j-simple', version: '2.0.0-alpha5'
44     implementation group: 'org.apache.logging.log4j', name: 'log4j-core', version: '2.18.0'
45     implementation group: 'org.assertj', name: 'assertj-core', version: '3.8.0'
46     implementation group: 'org.hamcrest', name: 'hamcrest-all', version: '1.3'
47     implementation 'io.github.bonigarcia:webdrivermanager:5.5.3'
48     implementation 'com.github.javafaker:javafaker:1.0.2'
49
50 }
```

GUIA PARA ELABORAR UNA AUTOMATIZACIÓN WEB

Explicación breve de las dependencias mencionadas

Serenity BDD: Bibliotecas para pruebas de comportamiento con Serenity.

Cucumber: Para pruebas BDD con Cucumber.

JUnit: Para pruebas unitarias.

Apache POI: Para trabajar con documentos de Microsoft Office.

SLF4J: Para registros.

Log4j: Para registros.

AssertJ y Hamcrest: Para aserciones en pruebas.

WebDriverManager: Administrador de webdrivers para la descarga automática de los mismos.

JavaFaker: Permite usar la clase Faker para generación de datos de prueba aleatorio.

Test

```
52 test {  
53     useJUnit {  
54         include "**/*";  
55         gradle.startParameter.continueOnFailure = true  
56         testLogging.showStandardStreams = true  
57         systemProperties System.getProperties()  
58         systemProperty "cucumber.filter.tags", System.getProperty("cucumber.filter.tags")  
59     }  
60 }
```

useJUnit: Este bloque indica que estás configurando JUnit para ejecutar pruebas.

include "/*":** Esta línea especifica que todos los archivos (pruebas) dentro del proyecto deben incluirse para la prueba.

gradle.startParameter.continueOnFailure = true: Esta línea establece que Gradle continúe ejecutando pruebas incluso si algunas de ellas fallan.

testLogging.showStandardStreams = true: Esta línea permite que se muestren las corrientes estándar de salida y error durante la ejecución de la prueba.

systemProperties System.getProperties(): Esta línea establece las propiedades del sistema para las pruebas como las mismas propiedades del sistema del proceso de construcción de Gradle.

systemProperty "cucumber.filter.tags", System.getProperty("cucumber.filter.tags"): Esta línea establece una propiedad del sistema para las pruebas de Cucumber, probablemente utilizada para filtrar pruebas según etiquetas.