

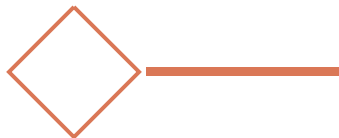


CURSO AUTOMATIZACION

Básico



Noviembre 2025



Contenido del Curso

- Contextualización: ¿Qué es la Automatización de Pruebas?
- ¿Cuáles son los Objetivos de Automatizar?
- ¿Dónde se puede aplicar Automatización?
- ¿Cuándo Automatizar y cuando NO Automatizar?
- ¿Cuáles son los Beneficios?
- ¿Qué debo considerar para el ROI (Return of Investment) de Automatización?
- Niveles de Automatización
- Herramientas de Automatización
- Locators
- Patrones (POM, PageFactory)
- Herramienta de Trabajo
- Katalon Recorder a Eclipse IDE
- Patrón POM (Page Object Model)
- Git
- Jenkins

Contextualización:

¿Qué es la Automatización de Pruebas?

Contextualización: ¿Qué es la Automatización de Pruebas?



En las pruebas de software, la **automatización** es aprovechada mediante el uso de herramientas de software especializadas, que permiten la ejecución de pasos y casos de prueba, a través de la comparación de resultados actuales con resultados esperados.

La automatización de pruebas utiliza un **método formal** que busca aumentar la eficiencia en los procesos de pruebas.

La automatización también es usada para automatizar procesos que apoyen las pruebas, tales como **extracción o generación de set datos**, entre otros.

¿Cuáles son los
Objetivos de
Automatizar?

¿Cuáles son los Objetivos de Automatizar?



Aumentar eficiencia

Optimizar procesos y reducir tiempos de ejecución



Incrementar cobertura

Ampliar el alcance de las pruebas realizadas



Minimizar riesgo

Reducir errores y fallos en producción



Reutilización

Aprovechar artefactos en múltiples ciclos



Mejora la Oportunidad

Cumplir iniciativas en tiempo y forma

¿Cómo puedo hacer mi trabajo de forma más eficiente?

¿Puedo usar automatización de pruebas para ello?

¿Dónde se puede aplicar
Automatización?

¿Dónde se puede aplicar Automatización?



Migración de datos

Automatización de procesos de transferencia y transformación de datos entre sistemas



Monitoreo en producción

Supervisión continua y automatizada del estado y rendimiento de aplicaciones en vivo



Generación de datos

Creación automática de conjuntos de datos de prueba para validación de sistemas



Procesos de configuración necesarios

Automatización de tareas de configuración y preparación de entornos de prueba

¿Cuándo Automatizar?

¿Cuándo Automatizar?

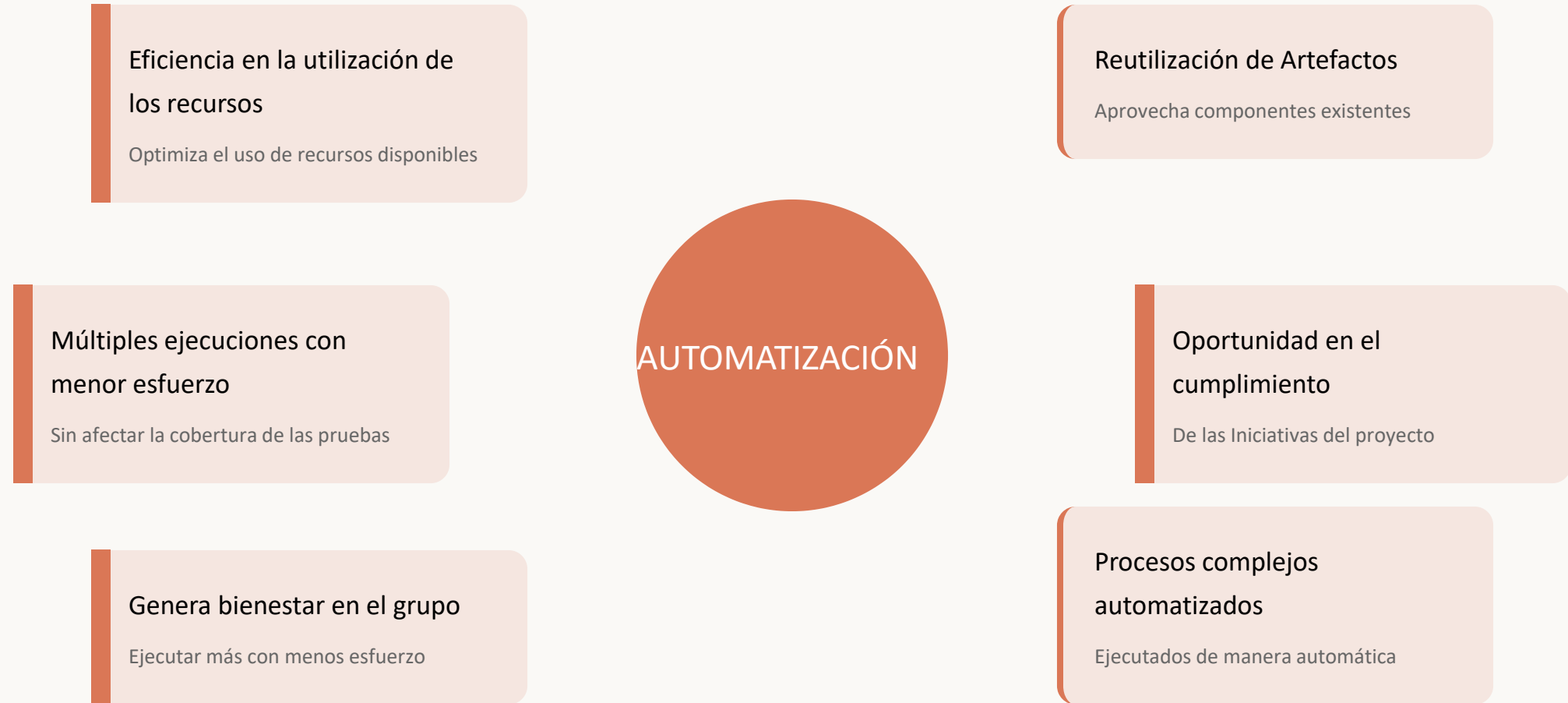
- Aplicaciones críticas que impacten el negocio
- Alta frecuencia en recepción de versiones de software
- Aumentar la cobertura para minimizar riesgo
- Aumentar la eficiencia en los procesos
- Tareas repetitivas que intervienen en el proceso de pruebas
- Pruebas que en el proceso de ejecución requiere verificar múltiple información

Cuándo NO Automatizar

- Pruebas exploratorias, pruebas aleatorias o de dominio específico.
- Procesos que se ejecutan una única vez, esto varía en el caso que se repita para cada proyecto.
- Procesos sin resultados predecibles, por tener un alto grado de análisis y decisión sobre los resultados obtenidos.
- Procesos que por su complejidad demandan una inversión de tiempo y dinero irrecuperable al automatizarlo.

¿Cuáles son los Beneficios?

¿Cuáles son los Beneficios?



¿Qué debo considerar para el ROI
(Return of Investment)
de Automatización?

ROI - Momento 1

Se debe considerar las características de los procesos manuales a ser automatizados, en la que se incluye:

- **Tiempo de ejecución del proceso de forma manual:** Considerar a nivel de detalle, para no caer en una medida errónea. Ejemplo, en un proceso de prueba, ejecutar un flujo toma 10 minutos de forma manual, esto considera las verificaciones y la respectiva toma de evidencia, ordenación y otros. No solo es ejecutar el proceso sin ninguna consideración.
- **El alcance del proceso automatizado tiene que estar claro:** Ejemplo, en el caso de ejecutar el proceso automatizado se necesite invertir varias horas en la generación de datos, este último no tiene que involucrarse en el análisis porque pertenece a otro proceso "generación de datos" y se debe trabajar de forma independiente o de manera que se pueda distinguir el porqué.
- **Tener claro la repetitividad y/o la frecuencia de ejecución:** En un tiempo determinado (finito), que permita obtener información concisa del ROI.
- **Estimar y posteriormente tomar los datos de tiempos de ejecución automatizada:** Del proceso, para poder diagramar la tendencia y comportamiento del ROI.
- **Tener en cuenta que estos datos se pueden obtener de forma automática:** De la ejecución automatizada.

ROI - Momento 2

Análisis Comparativo

En este caso se tiene que analizar del ROI comparando los comportamientos de la ejecución automatizada actual vs la última ejecución automatizada, esto permite identificar si se están ejecutando procesos de optimización de la automatización.

En resumen, es una comparación de lo automatizado vs lo automatizado hasta llegar a un comportamiento y frecuencia adecuadas.

Proyección del ROI

Asimismo, tener en cuenta que el ROI se puede proyectar para saber que se espera con respecto al proceso automatizado (**Antes de Automatizar**), y confirmar que la proyección esperada se cumplió o tuvo variaciones (**Después de Automatizar**) y si estos se alinean a la estrategia planteada cuando se optó la automatización del proceso.

Ejercicio Cálculo del ROI

Se tiene un aplicativo de nómina que consta de 4 funcionalidades.

Funcionalidad	Pantallas	Objetos	Casos de Prueba	Tiempo/Caso
Funcionalidad 1	6 pantallas	80 objetos	120 casos	7 minutos
Funcionalidad 2	3 pantallas	20 objetos	30 casos	8 minutos
Funcionalidad 3	2 pantallas	10 objetos	150 casos	5 minutos
Funcionalidad 4	3 pantallas	25 objetos	125 casos	6 minutos

Preparación de datos: La preparación de datos para cada una de las funcionalidades se demora en promedio 3 horas por cada una de ellas.

Por histórico se tiene que la ejecución manual de estos casos por funcionalidad se ha comportado así:

F1	F2	F3	F4
7 min	8 min	5 min	6 min

Ejercicio Cálculo del ROI

Esfuerzo estimado para automatizar las funcionalidades:

F1: 40 horas (3 min/ejecución)

F3: 6 horas (1 min/ejecución)

F2: 12 horas (2 min/ejecución)

F4: 4 horas (1 min/ejecución)

Si sabemos que el costo de un analista de pruebas Manual es de **US \$30 hora** y de un Automatizador **US \$50 hora**.

Tareas del ejercicio:

- Realice la proyección de Tiempos y Costos Acumulados
- Verifique el gráfico de Horas Acumuladas Automatizadas vs Manual
- Verifique el gráfico del Costo Acumulado Automatización vs Manual
- Identifique en que ciclo de ejecución se alcanza el punto de equilibrio entre Automatización vs Manual

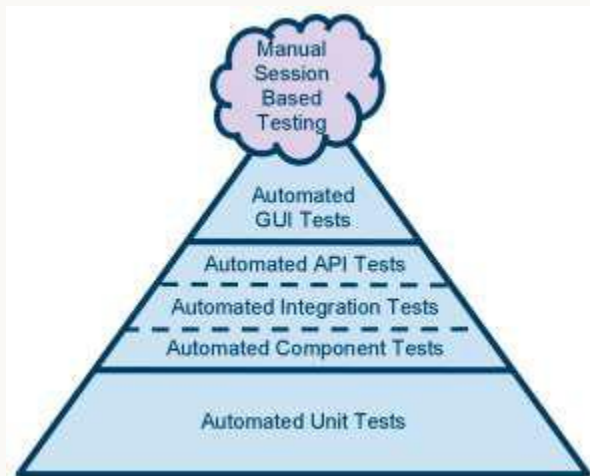
TIEMPO ESTIMADO DESARROLLO DEL EJERCICIO: **30 minutos**



Niveles de Automatización



Pirámide de Mike Cohn



UI

Interfaz Gráfica

Menos test de interfase gráfica automatizados

API

Servicios e Integración

Bastante test a nivel de API, integración de componentes, servicios

Unit

Test Unitarios

Muchos test unitarios automáticos

Un nivel estable de pruebas automáticas



Herramientas de Automatización

Herramientas de Automatización

Comerciales

UFT

Ranorex

Visual Test

SilkTest

RFT

Perfecto Mobile

Open Source

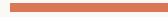
Selenium

Robot Framework

SoapUi

Xamarin

Sikuli



Locators

Identificación de los elementos de la interfaz

Es muy importante que dicho localizador se resuelva de forma única, para que la prueba sea correcta, y la acción no se realice sobre un elemento indeseado. Por otra parte, también debe tenerse en cuenta que el identificador elegido sea reutilizable en el futuro: por ejemplo debe evitarse la elección de un identificador que cambie con cada nueva versión de la aplicación.

Por ello, es muy importante elegir de manera adecuada los identificadores que vamos a usar en cada caso:

- **id:** Es la mejor opción siempre y cuando la página HTML tenga definidos correctamente los identificadores de sus elementos.
- **name:** Por definición, el atributo 'name' de un elemento HTML no tiene que ser único, con lo que el uso de este método de localización no garantiza que la prueba se ejecute de la manera deseada.
- **dom:** Esta opción utiliza el DOM de la página para hacer referencia a los elementos. El problema que presenta es que cualquier introducción de un nuevo elemento, o reorganización de los existentes, provoca que las referencias cambien, lo que puede invalidar las pruebas grabadas anteriormente.
- **xpath:** Este método de identificación es similar al anterior, pero en este caso hace uso de la estructura XML que posee todo documento HTML, para así hacer referencia a los elementos mediante una ruta, ya sea absoluta (partiendo desde el elemento /) o relativa (partiendo de un elemento conocido).

Identificación de los elementos de la interfaz

LOCATORS - Parte 2

link

Este método es el más utilizado al querer localizar un enlace. Su uso requiere conocer el texto que va a mostrar dicho enlace en la página HTML, por lo tanto no es útil en caso de existir enlaces con el mismo texto (devolvería el primero de ellos), enlaces sin texto, o enlaces con texto dinámico.

CSS

Este localizador consiste en identificar los elementos por sus propiedades de CSS. Este método tampoco garantiza la unicidad del elemento referenciado.

Tabla Comparativa de Locators

Localizador	Compatible Ext JS	Compatible JSF (JavaServer Face)	Independencia de la estructura HTML	Único Localizador
id	No	Si	Si	Si
name	No	Si	Si	No
identificador	No	Si	Si	No
dom	Si	Si	No	No
xpath	Si	Si	No	Si
link	Si	Si	Si	No
css	Si	Si	Si	No

POM (Page Object Model)

Es un patrón de diseño de objeto en Selenium, donde las páginas web se representan como clases, y los diversos elementos de la página se definen como variables en la clase.

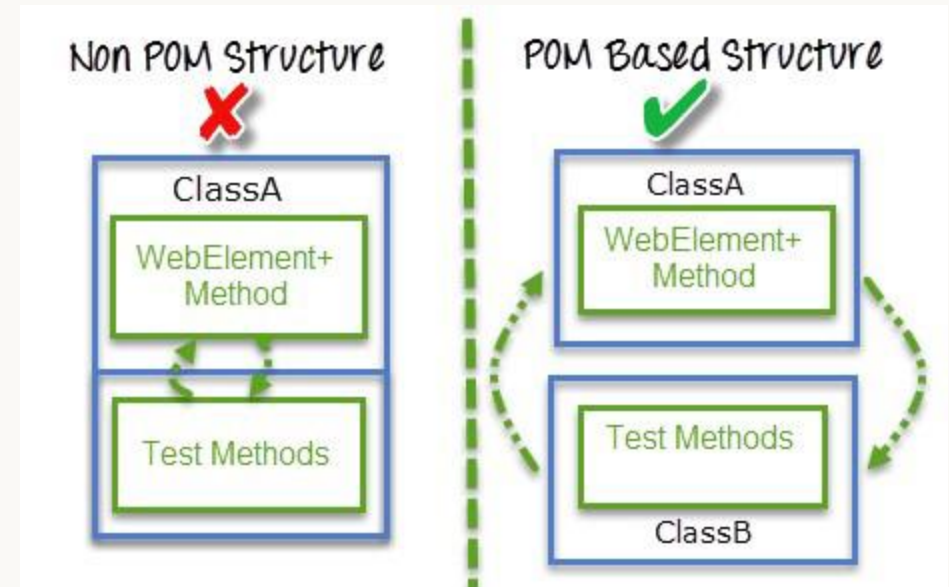
- **Páginas como clases:** Cada página web es una clase
- **Elementos como variables:** Los elementos se definen como variables
- **Interacciones como métodos:** Las acciones se implementan como métodos

Ejemplos de métodos:

```
clickLoginButton();
```

```
setCredentials(user_name, user_password);
```

Los métodos bien nombrados son fáciles de leer, funcionando como una forma elegante de implementar rutinas de prueba legibles y más fáciles de mantener o actualizar.



PageFactory

Page Factory es un concepto de **Página Modelo de Objetos incorporado** para Selenium WebDriver pero está muy optimizado.

Aquí también, seguimos el concepto de separación de Page Object Repository y Test Methods. Además, con la ayuda de la clase PageFactory, utilizamos las anotaciones **@FindBy** para encontrar WebElement.

Usamos el método **initElements** para inicializar elementos web.

@FindBy puede aceptar como atributos:

- tagName
- name
- id
- className
- partialLinkText
- linkText
- css
- xpath

WebElements are identify by
@FindBy Annotation

static initElements method of
PageFactory class for
initializing WebElement

```
@FindBy(xpath="//table//tr[@class='heading3']")
WebElement homePageUserName;

public Guru99HomePage(WebDriver driver){
    this.driver = driver;
    //This initElements method will create all WebElements
    PageFactory.initElements(driver, this);
}
```



Herramienta de Trabajo

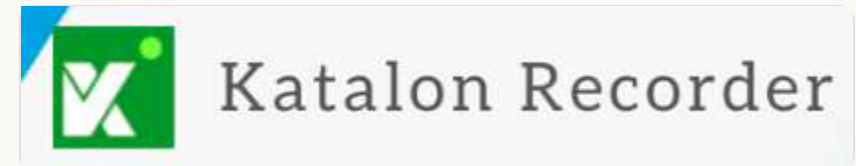


KATALON RECORDER

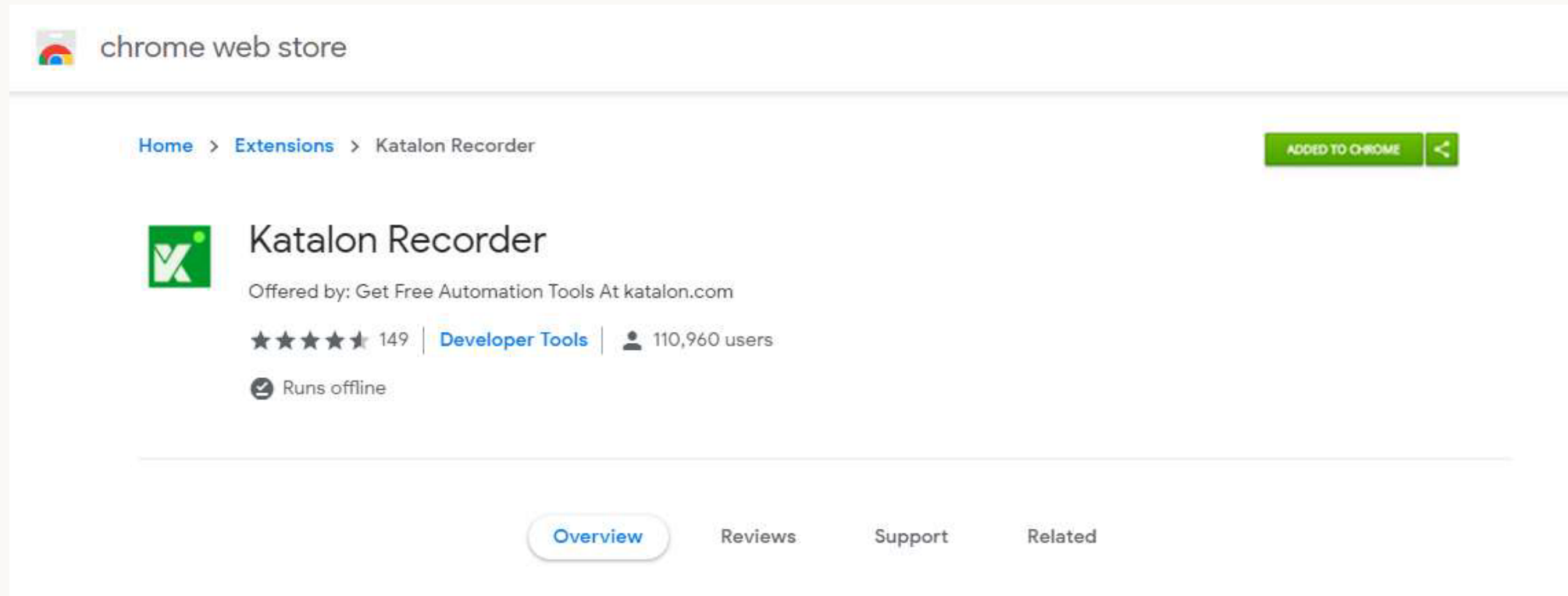
Katalon Recorder es una **extensión para Chrome y Firefox** que facilita la automatización de pruebas web.

Características principales:

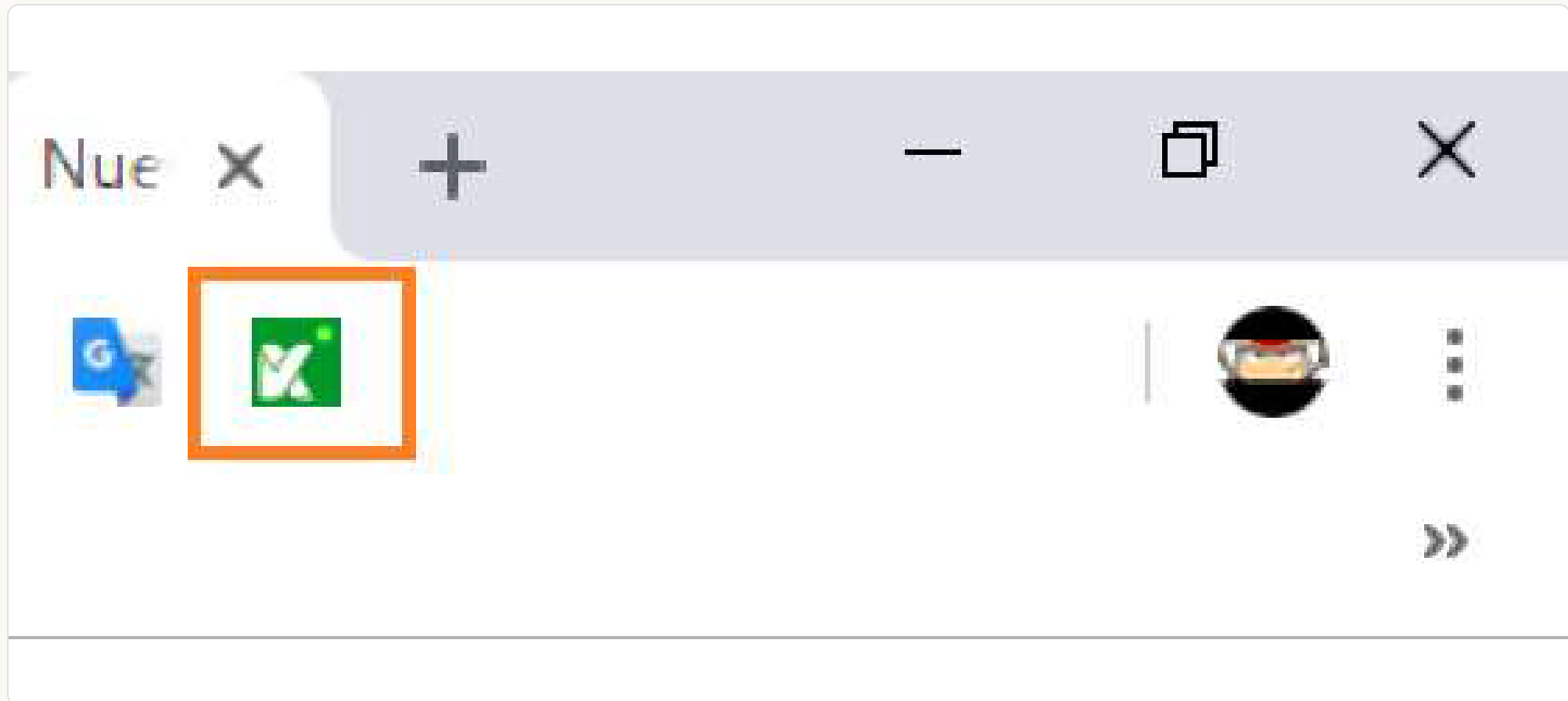
- Herramienta complementaria de **Katalon Studio**, una de las mejores herramientas de automatización
- **Capacidad para generar locators** ID, NAME y XPath automáticamente
- Constructores de localizadores integrados y personalizados
- Interfaz simple pero con ventajas ocultas poderosas



Agregar Katalon Recorder



Interfaz Katalon Recorder



Barra de Herramientas

Botón	Descripción
New	Crear un nuevo caso de prueba o conjunto de pruebas
Record	Para grabación de prueba de automatización
Play	Ejecutar caso de prueba individual seleccionado
Play Suite	Ejecutar el conjunto de pruebas seleccionado
Play All	Ejecutar todas las suites de prueba
Pause / Resume	Pausar y reanudar la ejecución actual
Stop	Detener la grabación o la ejecución actual
Export	Exportar el conjunto de prueba / caso de prueba actual a varios lenguajes de scripting y framework
Adjust play speed	Ajustar la velocidad de ejecución
Settings	Configuración de puertos para usuarios de Katalon Studio. Permitir cambiar el puerto predeterminado para comunicarse con el navegador activo
Help	Katalon Automation Recorder Guía del usuario

Ejemplos Katalon Recorder

Ejercicio 1 – Acciones Comunes

URL del Ejercicio:

<https://katalon-test.s3.amazonaws.com/demo-aut/dist/html/form.html>

First name	<input type="text"/>
Last name	<input type="text"/>
Gender	<input type="radio"/> Male <input type="radio"/> Female <input type="radio"/> In Between
Date of birth	<input type="text"/>
Address	<input type="text"/>
Email	<input type="text"/>
Password	<input type="text"/>
Company	<input type="text"/>
Role	Developer ▼

Ejercicio 1 – Acciones Comunes

Objetivos del Ejercicio:

- 1 Definir velocidad de ejecución de Caso
- 2 Verificar los componentes de un paso (Command, Target y Value)
- 3 Agregar un paso de manera manual
- 4 Seleccionar el Target del elemento en el paso manual
- 5 Verificar que ese Target si se identifique en pantalla
- 6 Agregar un paso Automático

Ejercicio 2 – Capturas de Pantalla

URL DE PRÁCTICA

<https://katalon-test.s3.amazonaws.com/demo-aut/dist/html/form.html>

OBJETIVO

Agregar paso para captura de Pantalla con el comando "captureEntirePageScreenshot"

En el **Target** se asigna el nombre de la Imagen

Command	Target	Value
open	https://katalon-test.s3.amazonaws.com/demo-aut/dist/html/form.html	
click	id=comment	
captureEntirePageScreenshot	before-adding-comment	
type	id=comment	Added by Alex.
captureEntirePageScreenshot	after-adding-comment	

Ejercicio 3.1 - Trabajar con variables

URL DE PRÁCTICA

<https://katalon-test.s3.amazonaws.com/demo-aut/dist/html/form.html>

Objetivo:

- Almacenar en la variable "Comentario" un valor
- Asignar a un objeto en pantalla tipo **Textbox** el valor de dicha variable

Command	Target	Value
open	https://katalon-test.s3.amazonaws.com/demo-aut/dist/html/form.html	
store	Alex	Comentario
type	id=comment	\${Comentario}

Ejercicio 3.2 - Trabajar con variables

<https://katalon-test.s3.amazonaws.com/demo-aut/dist/html/form.html>

Objetivo:

- Almacenamos valores en dos variables "a" y "b"
- Realizamos la suma de estas dos variables "a" y "b" en una tercera "c"
- Asignamos a un campo tipo Textbox el valor de la variable "c"

Command	Target	Value
open	https://katalon-test.s3.amazonaws.com/demo-aut/dist/html/form.html	
storeEval	1	a
storeEval	2	b
storeEval	a + b	c
type	id=comment	\$(c)

Ejercicio 4 – Verificar

Objetivo:

Verificar que se presente el mensaje de Campo requerido cuando se deja sin diligenciar.

First name	<input type="text" value="Yeison"/>
Last name	<input type="text" value="Arias"/>
Gender	<input checked="" type="radio"/> Male <input type="radio"/> Female <input type="radio"/> In Between
Date of birth	<input type="text" value="05/02/1981"/>
Address	<input type="text"/>

This field is required.

Ejercicio 5 – Ciclo While

Objetivo:

Realizar ejercicio con iteración While

Command	Target	Value
open	https://katalon-test.s3.amazonaws.com/demo-aut/dist/html/form.html	
store	1	ii
while	ii<5	
type	id=first-name	\$(ii)
storeEval	\$(ii)+1	ii
endWhile		

Ejercicio 6 – Test Data

URL de Práctica:

<https://katalon-test.s3.amazonaws.com/demo-aut/dist/html/form.html>

Objetivo:

Realizar diligenciamiento con archivo CSV

Log
Screenshots
Variables
Data Driven

+ Add CSV File
+ Add JSON File

Type	Name	Actions
CSV	Libro1.csv	✎ 🗑

type	id=password	\$(password)
::endLoadVars		

Command	Target	Value
loadVars	Libro1.csv	
:: open	https://katalon-test.s3.amazonaws.com/demo-aut/dist/html/form.html	
type	id=first-name	\$(firstname)
type	id=last-name	\$(lastname)

Ejemplos Adicionales



Katalon Recorder

Repositorio oficial con ejemplos prácticos de Katalon Recorder

REPOSITORIO GITHUB

<https://github.com/katalon-recorder/katalon-recorder-samples>

Accede a casos de prueba de ejemplo, scripts y recursos para practicar con Katalon Recorder



Selenium



¿Qué es Selenium?

Selenium es un entorno de pruebas que se utiliza para comprobar si el software que se está desarrollando funciona correctamente.

Esta herramienta permite:

- Grabar, editar y depurar casos de prueba que se pueden automatizar
- Editar acciones o crearlas desde cero
- Ayuda en las pruebas de regresión mediante pruebas automatizadas reutilizables



Breve Historia

- Comenzó a desarrollarse en 2004 por Jason Huggins y poco a poco se fueron uniendo varios especialistas.
- Un gran detalle es que este software es de código abierto (bajo licencia Apache 2.0) y puede ser descargado y usado sin pagar. Si te estás preguntando "¿dónde puedo comprar Selenium? o ¿dónde puedo descargar Selenium?", solamente tienes que entrar en la siguiente para descargar esta herramienta de manera gratuita.

Descarga gratuita:

<https://www.selenium.dev/downloads/>

- Como curiosidad indicar que el significado de Selenium surgió a raíz de un comentario que Jason Huggins le hizo a un competidor de aquella época llamado Mercurio: *el envenenamiento por mercurio se puede curar con Selenio*. Esto provocó que la mayoría de personas que lo escucharon empezasen a llamar al proyecto de Huggins, "Selenium".

¿Para qué sirve Selenium?

OBJETIVO PRINCIPAL

Comprobar que un software funcione correctamente

Usuarios de Selenium

- **Programadores QA:** Profesionales especializados en asegurar la calidad del software mediante pruebas automatizadas
- **Engineer QA:** Ingenieros que no solo comprueban el funcionamiento correcto, sino que también trabajan en la prevención de problemas futuros

La mejor característica de Selenium para muchos Ingenieros QA es su capacidad de **evitar problemas futuros**, no solo detectar los existentes

¿Cuáles son las opciones de Selenium?

- Selenium IDE
- Selenium Remote Control
- Selenium WebDriver
- Selenium Grid



¿Qué es Selenium IDE?

Selenium IDE permite editar, grabar y depurar lo que se muestra en el navegador. Ha sido diseñado como una extensión de navegador con soporte en Firefox y Chrome.

Características principales:

- Herramienta de línea de comandos para integración de servidores o ejecuciones de pruebas paralelas
- Configuración de puntos de ruptura
- Funciones de depuración
- Soporte para integración de terceros
- Reutilizar casos de pruebas



Selenium Remote Control

Remote Control (RC) es un sistema cliente/servidor que permite utilizar el navegador web de forma local o en otro ordenador.

El funcionamiento del Selenium RC se basa en sus componentes. Cada uno de los componentes tiene un papel que desempeñar en la ejecución de los scripts de prueba.

Los componentes de Selenium RC son responsables de la automatización de los scripts de prueba:

Servidor Selenium

- Responsable de iniciar y eliminar los navegadores
- Interpreta y ejecuta los comandos de Selenium que se envían desde el programa de prueba
- Actúa como proxy HTTP, interceptando y verificando los mensajes HTTP que se pasan del AUT al navegador

Bibliotecas Cliente

Existen bibliotecas cliente que ofrecen una interfaz entre el servidor Selenium y cada uno de los lenguajes de programación.

Selenium WebDriver - Parte 1

EVOLUCIÓN DE SELENIUM

Sucesor de Remote Control

Selenium WebDriver es la evolución natural de Selenium Remote Control, ofreciendo una arquitectura más simple y eficiente

- **No requiere servidor especial:** A diferencia de Remote Control, WebDriver inicia directamente una instancia del navegador y la controla sin necesidad de un servidor intermediario
- **Permite uso local o remoto:** Flexibilidad para ejecutar pruebas en el navegador local o en máquinas remotas según las necesidades del proyecto

Lenguajes de Programación Compatibles

PHP

Java

.Net

Perl

Ruby

Python

No es necesario dominar todos los lenguajes, con desenvolverse bien con uno solo es más que suficiente

Selenium WebDriver - Parte 2

Ventajas técnicas de Selenium WebDriver

- **Arquitectura más simple que Remote Control**

La arquitectura de WebDriver es más simple que la de Remote Control, lo que facilita su implementación y mantenimiento sin comprometer la funcionalidad

- **Más rápido: habla directamente al navegador**

WebDriver es más rápido que RC, ya que habla directamente al navegador sin necesidad de intermediarios, reduciendo la latencia en la ejecución de pruebas

- **Utiliza el motor del propio navegador**

Lo mejor de todo es que utiliza el motor del propio navegador que se está utilizando en ese momento, garantizando compatibilidad y comportamiento nativo

- **Interacción realista con elementos de la página**

Interactúa con los elementos de la página de manera más realista. Por ejemplo, si aparece un cuadro de texto deshabilitado en la página web, una persona real no podría escribir en dicho cuadro y WebDriver actuaría así, mientras que otras herramientas no actuarían de la misma manera

¿Qué es Selenium Grid?

Existen dos razones principales para utilizar Selenium Grid

1

Ejecutar pruebas con **múltiples navegadores** a la vez (con diferentes versiones si es necesario) y con **diferentes sistemas operativos**, de tal manera que puedes utilizar, por ejemplo, Google Chrome a la vez en Windows, iOS, Android, etc.

2

Selenium Grid puede **reducir ampliamente el tiempo** que tarda un paquete de pruebas en completarse. Si se quiere realizar un conjunto de 200 pruebas pongamos hipotéticamente que se tardan 8 horas en comprobarlo todo al 100%, al utilizar 4 máquinas diferentes con Selenium Grid veremos su tiempo reducido y aproximadamente tardará una cuarta parte del total requerido en un inicio.

Ejemplo: 200 pruebas en 8 horas → **2 horas** con 4 máquinas

Gracias

CURSO AUTOMATIZACIÓN BÁSICO

Java y Selenium
