# Introduction

## What is GWLST?

Groovy WebLogic Scripting Tool is a [domain-specific language](http://en.wikipedia.org/wiki/Domain-specific_programming_language) embedded in the [Groovy](http://groovy.codehaus.org/) programming language that uses Groovy dynamic language features to seamlessly access the management interface of a remote WebLogic Server instance. By connecting to an admin server, GWLST can be used to control domain configuration and runtime operations and statistics for the entire domain.

WebLogic Server already exposes its configuration and runtime management interfaces over [JMX](http://docs.oracle.com/javase/6/docs/technotes/guides/jmx/index.html), and just about anything that can be done through WebLogic Console and more can also be done programmatically over this interface. GWLST merely makes this far more convenient than other techniques including Oracle’s [WLST](http://docs.oracle.com/cd/E21764_01/web.1111/e13715/intro.htm).

GWLST can be used as a scripting tool to update domain configuration or perform operations tasks instead of or in conjunction with using the WebLogic Console. It can also be used as a class library to develop standalone tools.

## How does GWLST compare to WLST?

### GWLST requires a running server

WLST offers both an offline mode that operates on files and an online mode that connects to a running server. WLST Offline manipulates a WebLogic Server domain directory and configuration files without a running server, and it remains one of the tools used to create a domain from scratch.

GWLST plays the same role as WLST Online in that it acts as a JMX client to a running WebLogic Server.

### GWLST is self-contained and remote

GWLST runs remotely from the WebLogic Server installation it connects to, and it does not require a full WebLogic Server installation where it runs. It does not require a complex shell script launch.

### GWLST is thin

The coupling between GWLST and WebLogic Server is very small, which makes GWLT itself backwards compatible to WebLogic Server 9.0 or so.

### Groovy instead of Jython

Whereas GWLST is embedded in Groovy, WLST is built in [Jython](http://www.jython.org/). Although a matter of taste and preference, GWLST scripts tend to be about half as long as the corresponding WLST script. Groovy also integrates seamlessly with Java, and you can write classes in GWLST that you can call from Java.

## How does scripting in GWLST compare to programming in Java?

### Dynamic language features

Ironically, the static nature of the Java programming language is not very fit for the dynamic nature of Java Management Extensions (JMX). When a Java caller accesses an MBean, its static interface is not known at compile time, and without that the Java program must use the verbose reflection style rather than the native “dots and parenthesis”. While this may seem cosmetic for small programs, it adds significant noise.

The Groovy programming language has dynamic language features that allow us to access the remote MBeans just as if they were ordinary objects.

### Groovy instead of Java

Groovy is in itself more expressive and better suited for scripting than Java. Unlike Java, Groovy/GWLST scripts can run “on the fly” as if they were interpreted. At the same time, Groovy integrates seamlessly with Java since it compiles into Java bytecode, runs on the same JVM and uses the same libraries. It is therefore possible to write a tool in Groovy with GWLST as a class library, compile and package as a JAR (embedding the Groovy runtime) without anyone knowing it’s not “Java”.

### Brevity

Comparing a GWLST script with the corresponding Java program, you will find the latter is about 10 times longer.

## Example

# Running GWLST