Selenium Grid

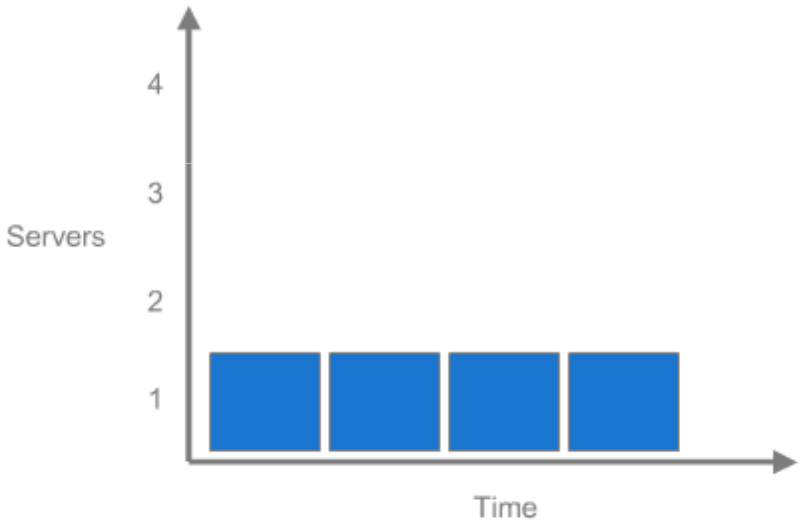
Don’t wait so long for tests results

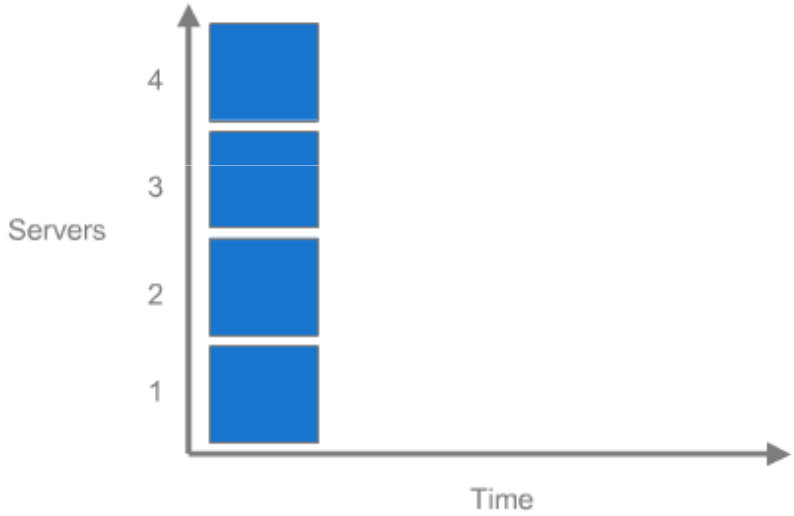
Selenium Grid

STOP WAITING!

- Distribute your tests on multiple machines
- Run your tests in parallel
- Dramatically speeds up in-browser web testing

Selenium Grid: What is faster?

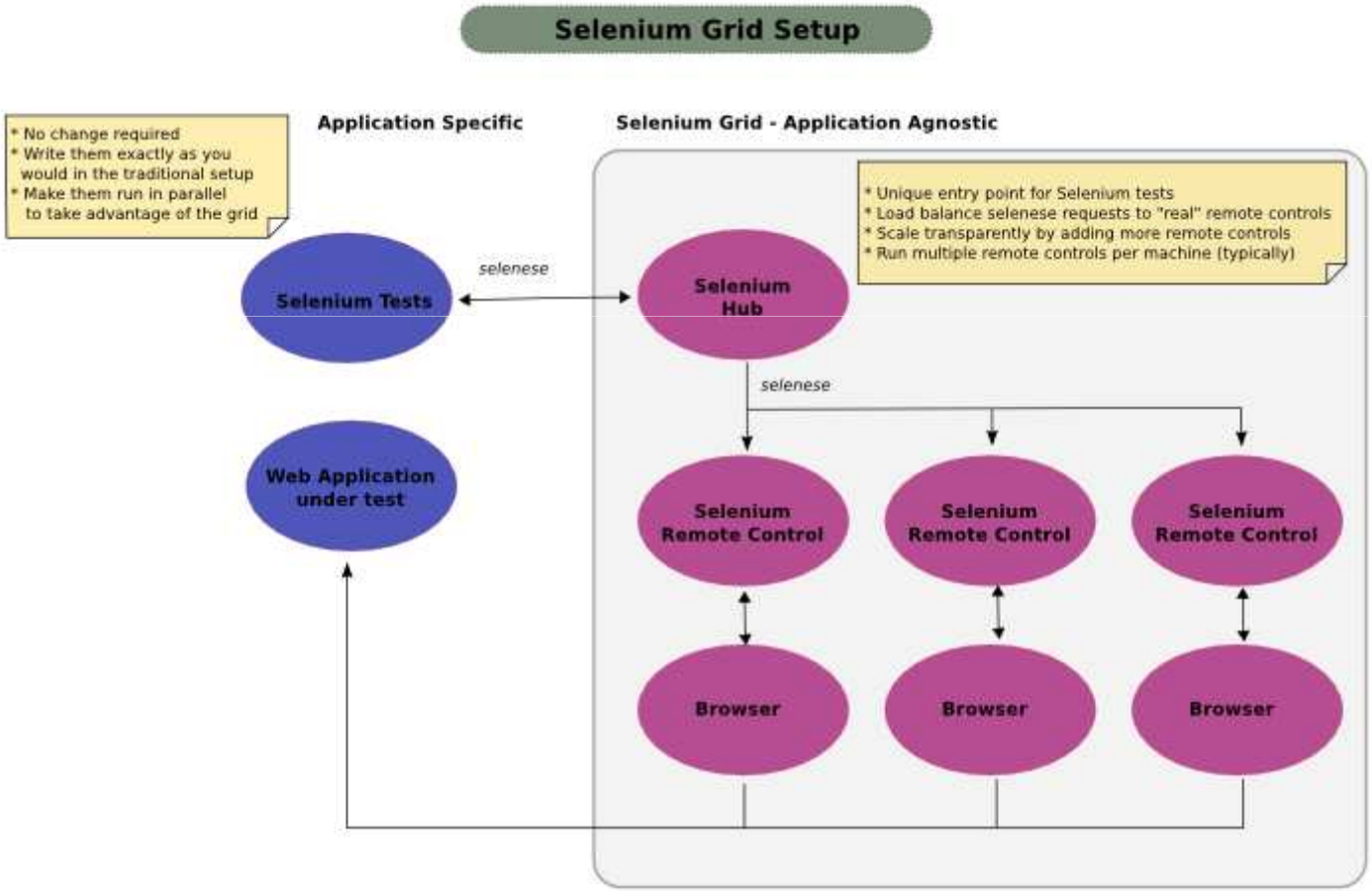




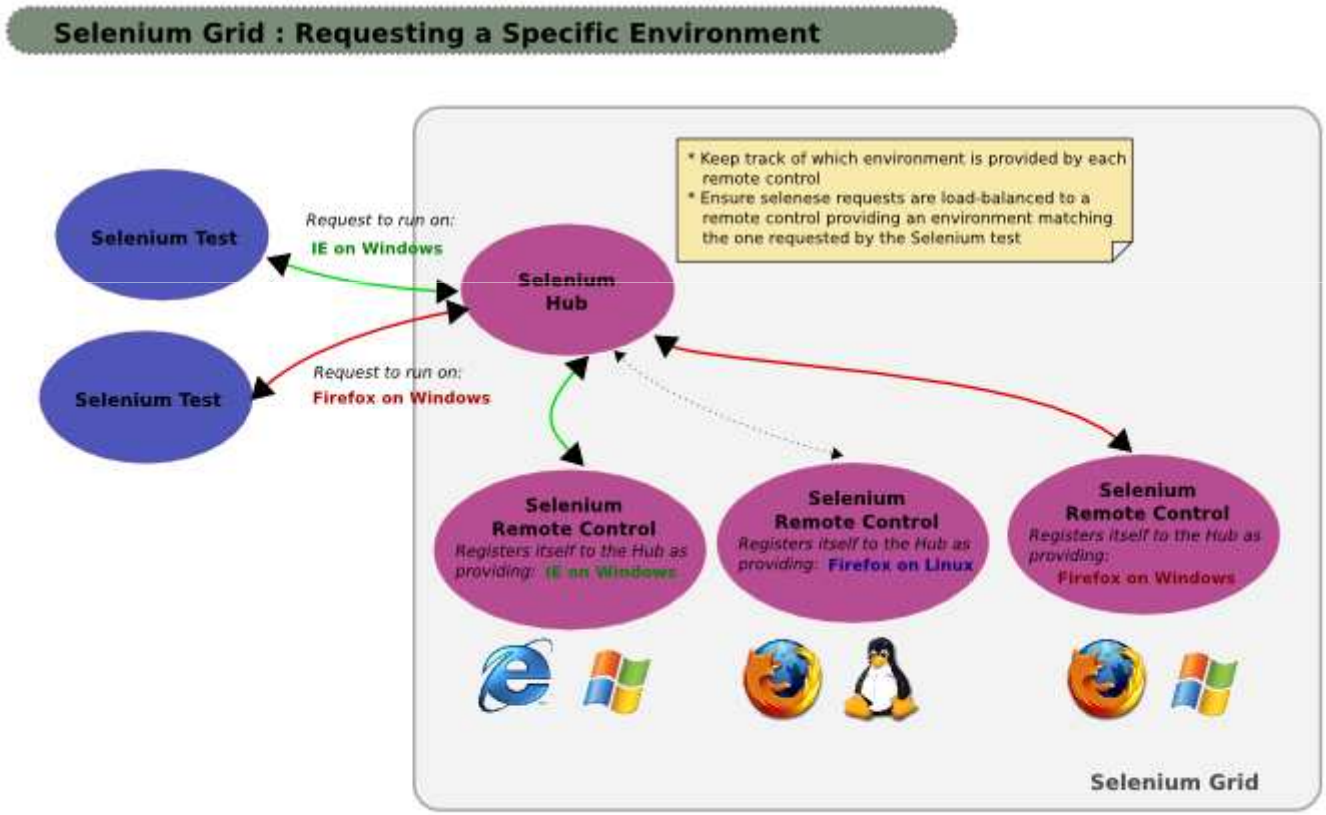
Selenium Grid: Philosophy

- One node = One browser = One environment
  - Simplify browser reservation process
  - Easy to reschedule tests
  - More than one IE on the same node doesn’t work because IE modifies registry settings and configure LAN directly
  - Node should be restarted from time to time
  - Each node can provide specific environment
- Tests can run on local machine or Grid without changes
  - Easy to write and test
  - Test isolation level depends on application

Selenium Grid: Parallel Execution



Selenium Grid: Some Environments



Selenium Grid: Setup

- Launch hub
  - *java -jar selenium-server-standalone-${VERSION}.jar -role hub*
  - Open grid console (*http://localhost:4444/grid/console*)
  - Configure with YAML (*-grid1Yml XXX.yml*) or JSON (*-hubConfig XXX.json*)
- Start nodes pointing them to the hub
  - *java –jar selenium-server-standalone-${VERSION}.jar –role rc*
  - *java –jar selenium-server-standalone-${VERSION}.jar –role webdriver*
- Run tests
  - Use RemoteWebDriver pointing to hub or usual Selenium RC

```
DesiredCapabilities capabilities = DesiredCapabilities.firefox();
capabilities.setVersion("6.0");
URL hubURL = new URL("http://localhost:4444/wd/hub");
WebDriver driver = new RemoteWebDriver(hubURL, capabilities);
```

  - TestNG parallel configuration
  - Parallel JUnit tests via Maven or with some external libraries

Selenium Grid: Node Configuration

- **port** – port that the node will be listening at, must be unique on the machine the nodes runs on
  - *-port 5556*
- **hub** – which hub the node should register/unregister to
  - *-hub http://localhost:4444/grid/register*
- **browser** – what browser does node support
  - *-browser browserName=firefox,version=3.6,platform=LINUX*
  - *-browser browserName=firefox,version=3.6,maxInstances=5*
  - *by default node will start with 5 firefox, 5 chrome, and 1 internet explorer*
- **nodeTimeout** – timeout in seconds before hub releases node with no commands to it
  - *-nodeTimeout 60*
- **maxSession** – maximal number of browsers run in parallel on the same node
  - *-maxSession 5*
- **registerCycle** – how often node must register itself
  - *-registerCycle 5000*

Selenium Grid: Not Enough Power?

- Amazon EC2
  - ***Elastic*** – one, hundreds or thousands of server instances for minutes, hours or months
  - ***Completely Controlled***
  - ***Flexible*** – multiple OS, memory, processor and storage configuration
  - ***Reliable*** - the Amazon EC2 SLA commitment is 99.95% availability for each Amazon EC2 Region
  - ***Features for Building Failure Resilient Applications*** – multiple locations, elastic IP address, elastic block store
  - ***Secure*** – Amazon provides web service interfaces to configure firewall settings that control network access to and between groups of instances
  - ***Inexpensive*** - you pay a very low rate for the compute capacity you actually consume with division on on-demand instances and reserved instances

Selenium Grid: EC2

- Signup for Amazon Accounts (AWS and EC2)
- Create Web Service Access Keys and X.509 Certificate
- Verify That JDK 1.5 or Higher is Installed
- Download EC2 API Command Line Tools
- Setup your environment
- Check Your Installation
- Install Capistrano
- Install Selenium Grid
- Run your tests

Selenium Grid: EC2 commands

```
cap ec2:check_settings # Check EC2 related configuration.
cap grid:boot          # Boot EC2 Instances for a Selenium Grid of 1 Hub and ...
cap grid:info          # Display infoemation about current Selenium Grid
cap grid:shutdown      # Shutdown EC2 Instance used to run Selenium Hub.
cap grid:start         # Start Selenium Grid Hub.
cap grid:stop          # Stop Selenium Grid Hub.
cap hub:boot           # Boot a new EC2 Instance to Run Selenium Grid Hub.
cap hub:console        # Open Selenium Grid Hub Console in a browser.
cap hub:logs           # View Selenium Grid Hub logs.
cap hub:restart        # (Re)start Selenium Grid Hub.
cap hub:shutdown       # Shutdown EC2 Instance used to run Selenium Hub.
cap hub:start          # Start Selenium Grid Hub.
cap hub:stop           # Stop Selenium Grid Hub.
cap invoke             # Invoke a single command on the remote servers.
cap rc:boot            # Boot a new EC2 Instance to run a collection of Selen...
cap rc:restart         # (Re)start Remote Controls for all farms.
cap rc:shutdown        # Shutdown all EC2 Instances used to run Selenium Grid...
cap rc:start           # Start Remote Controls for all farms.
cap rc:stop            # Stop Remote Controls for all farms.
cap rc:view            # Visualize what is happening in Remote Control Farms ...
cap rc:vnc:start       # Start VNC server on all Remote Control Farms
```

Selenium Grid: Demo

- Configure and run hub and some nodes
- Run demo in sequence
- Run demo in parallel

Selenium Grid: Alternatives

- ***Sauce Labs*** – cloud-based service to run Selenium tests in parallel using Amazon S3 and EC2
- ***GridGain*** – open source Java cloud framework
- ***Continuous Integration server*** – Hudson, TeamCity or some other with multiple build execution

Questions & Answers