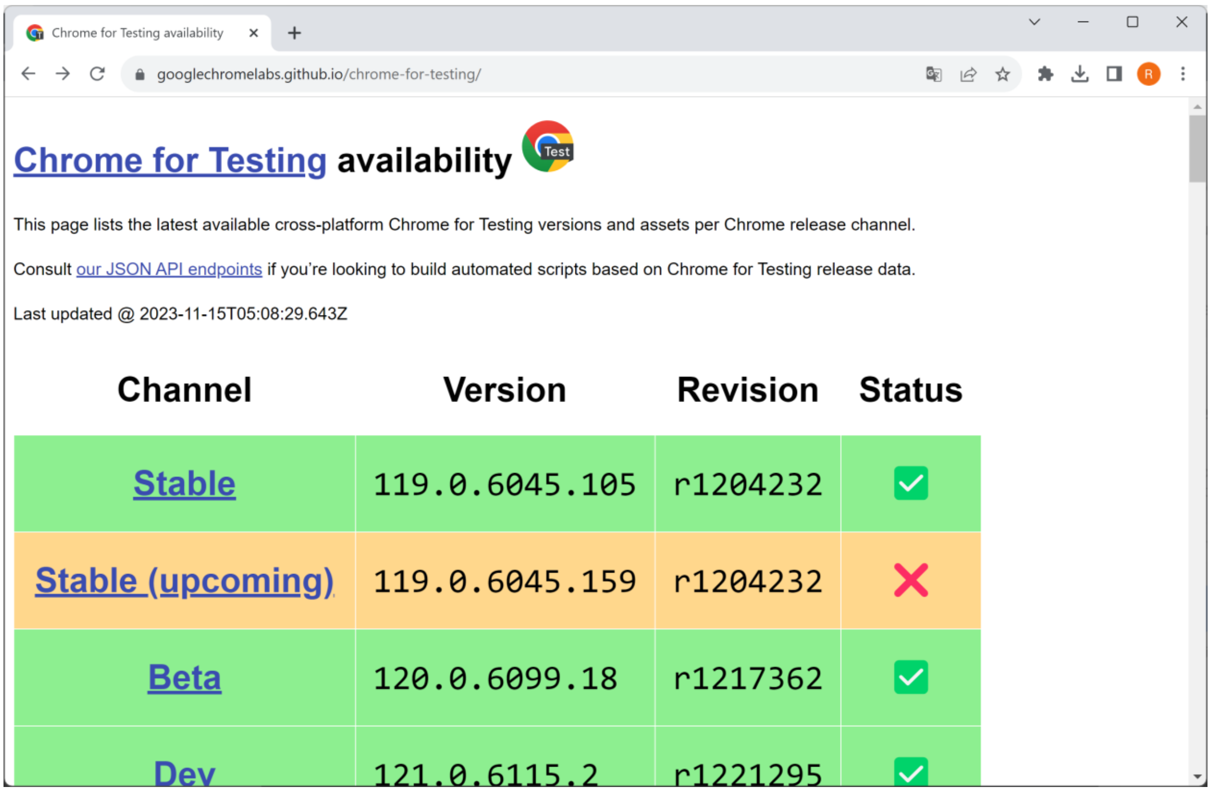
Lab 13

Part 1:

First go to the page https://googlechromelabs.github.io/chrome-for-testing/


Chrome for Testing availability
googlechromelabs.github.io/chrome-for-testing/

Chrome for Testing availability

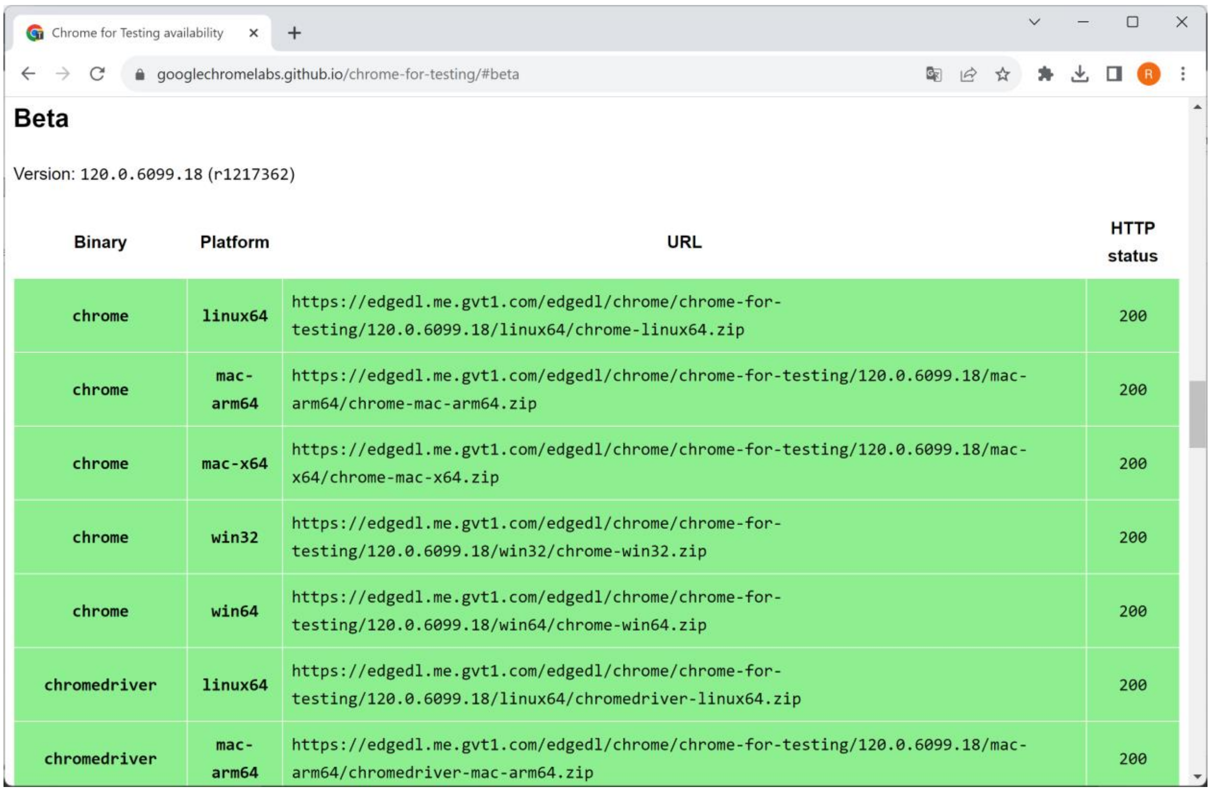
This page lists the latest available cross-platform Chrome for Testing versions and assets per Chrome release channel.

Consult our JSON API endpoints if you're looking to build automated scripts based on Chrome for Testing release data.

Last updated @ 2023-11-15T05:08:29.643Z

Channel	Version	Revision	Status
Stable	119.0.6045.105	r1204232	✓
Stable (upcoming)	119.0.6045.159	r1204232	✗
Beta	120.0.6099.18	r1217362	✓
Dev	121.0.6115.2	r1221295	✓

The go to the **Beta** part of the page


Chrome for Testing availability
googlechromelabs.github.io/chrome-for-testing/#beta

Beta

Version: 120.0.6099.18 (r1217362)

Binary	Platform	URL	HTTP status
chrome	linux64	https://edgedl.me.gvt1.com/edgedl/chrome/chrome-for-testing/120.0.6099.18/linux64/chrome-linux64.zip	200
chrome	mac-arm64	https://edgedl.me.gvt1.com/edgedl/chrome/chrome-for-testing/120.0.6099.18/mac-arm64/chrome-mac-arm64.zip	200
chrome	mac-x64	https://edgedl.me.gvt1.com/edgedl/chrome/chrome-for-testing/120.0.6099.18/mac-x64/chrome-mac-x64.zip	200
chrome	win32	https://edgedl.me.gvt1.com/edgedl/chrome/chrome-for-testing/120.0.6099.18/win32/chrome-win32.zip	200
chrome	win64	https://edgedl.me.gvt1.com/edgedl/chrome/chrome-for-testing/120.0.6099.18/win64/chrome-win64.zip	200
chromedriver	linux64	https://edgedl.me.gvt1.com/edgedl/chrome/chrome-for-testing/120.0.6099.18/linux64/chromedriver-linux64.zip	200
chromedriver	mac-arm64	https://edgedl.me.gvt1.com/edgedl/chrome/chrome-for-testing/120.0.6099.18/mac-arm64/chromedriver-mac-arm64.zip	200

Download both the chromedriver and chrome-headless-shell for you operating system.

Unzip the chrome driver somewhere on your filesystem (for example C:\tmp\chromedriver-win64)

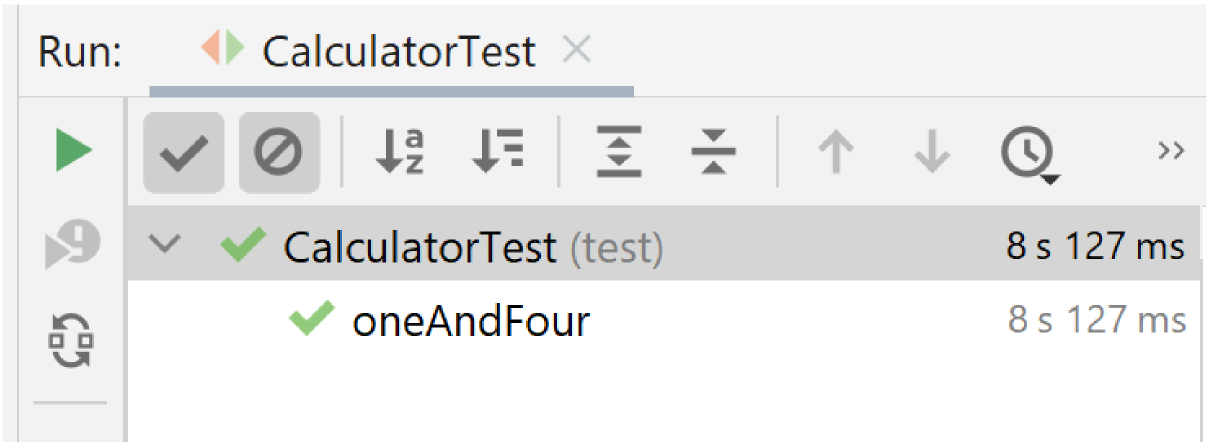
Unzip the chrome-headless-shell somewhere on your filesystem (for example C:\tmp\chrome-headless-shell-win64)

Given is the project **WebdriverProject**
In the folder **src/test/java/withoutpageobject** you find the file **CalculatorTest** that test the working of the online calculator at http://www.rekenmachine-calculator.nl/

Modify the file so that the chrome driver and the chrome headless shell points to the correct file location:

```
@Before
public void createWebDriver() {
   System.setProperty("webdriver.chrome.driver", "C:\\tmp\\chromedriver-
win64\\chromedriver.exe");
   ChromeOptions options = new ChromeOptions();
   options.setBinary("C:\\tmp\\chrome-headless-shell-win64\\chrome-headless-
shell.exe");
   options.addArguments("--remote-allow-origins=*");
   // create chrome instance
   driver = new ChromeDriver(options);
   driver.manage().timeouts().implicitlyWait(50, TimeUnit.SECONDS);
   driver.manage().timeouts().pageLoadTimeout(100, TimeUnit.SECONDS);
}
```

Run the test and if everything is correct you should see that the test passed:

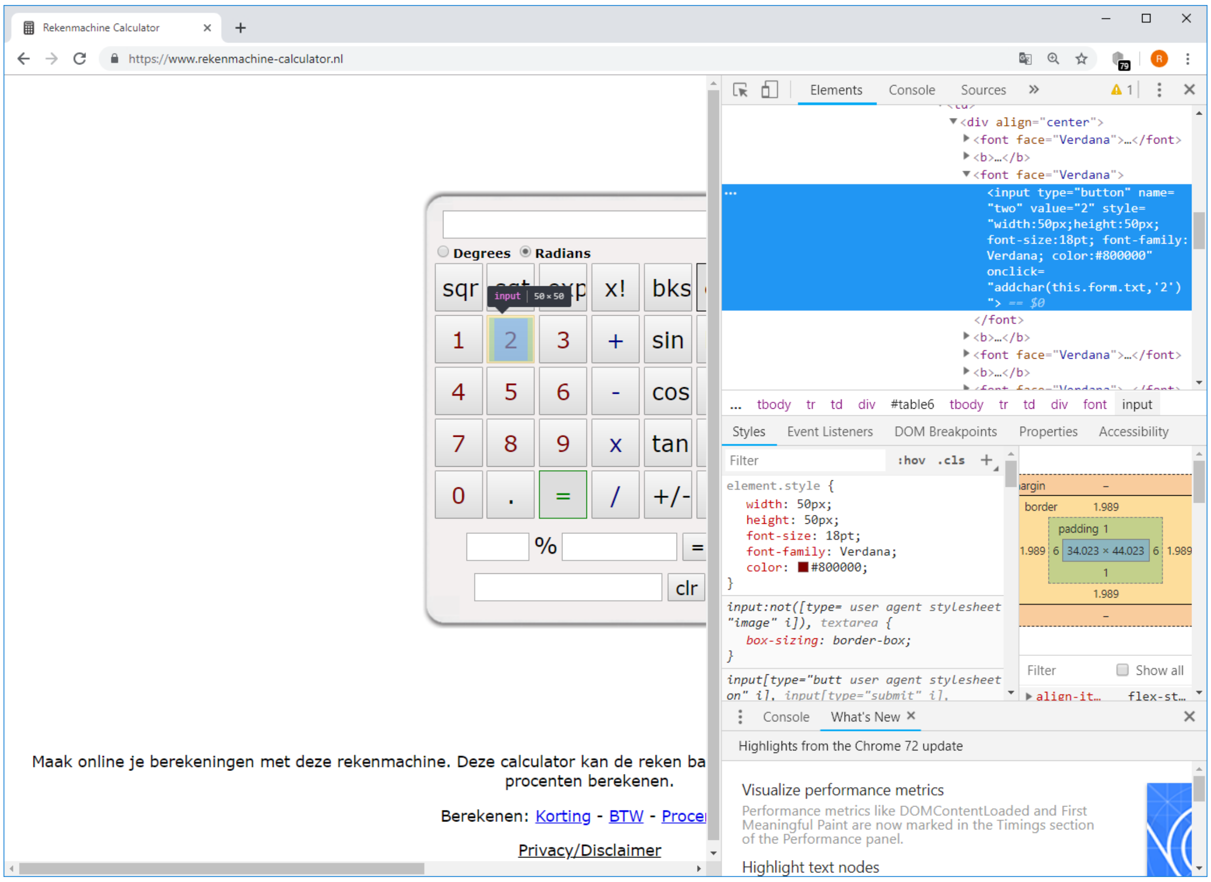


```
Run:   CalculatorTest ×

CalculatorTest (test)                8 s 127 ms
    oneAndFour                       8 s 127 ms
```

Add some more tests that test the correct working of the calculator webpage. If you need to find the locator of a certain web element you can do the following:

In the chrome browser, go to https://www.rekenmachine-calculator.nl/

Right-click a webelement (for example the button 2) and select **Inspect**.

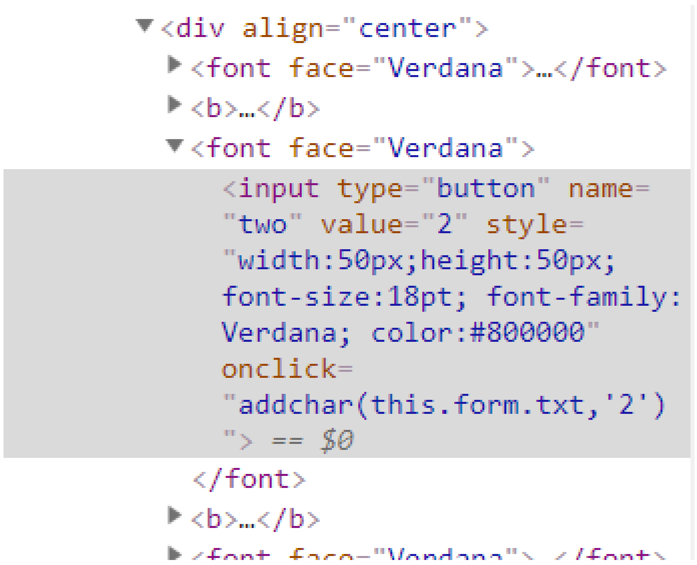


```
<div align="center">
  <font face="Verdana">…</font>
  <b>…</b>
  <font face="Verdana">
    <input type="button" name="two" value="2" style="width:50px;height:50px;
    font-size:18pt; font-family:Verdana; color:#800000" onclick=
    "addchar(this.form.txt,'2')"> == $0
  </font>
  <b>…</b>
  <font face="Verdana">…</font>
  <b>…</b>

… tbody tr td div #table6 tbody tr td div font input

element.style {
  width: 50px;
  height: 50px;
  font-size: 18pt;
  font-family: Verdana;
  color: ■#800000;
}
input:not([type="image" i]), textarea {    user agent stylesheet
  box-sizing: border-box;
}
```

Maak online je berekeningen met deze rekenmachine. Deze calculator kan de reken ba procenten berekenen.

Berekenen: Korting - BTW - Proce

Privacy/Disclaimer



```
▼<div align="center">
  ▶<font face="Verdana">…</font>
  ▶<b>…</b>
  ▼<font face="Verdana">
      <input type="button" name=
      "two" value="2" style=
      "width:50px;height:50px;
      font-size:18pt; font-family:
      Verdana; color:#800000"
      onclick=
      "addchar(this.form.txt,'2')
      "> == $0
    </font>
  ▶<b>…</b>
```

You see now that this button has the locator **name="two"**

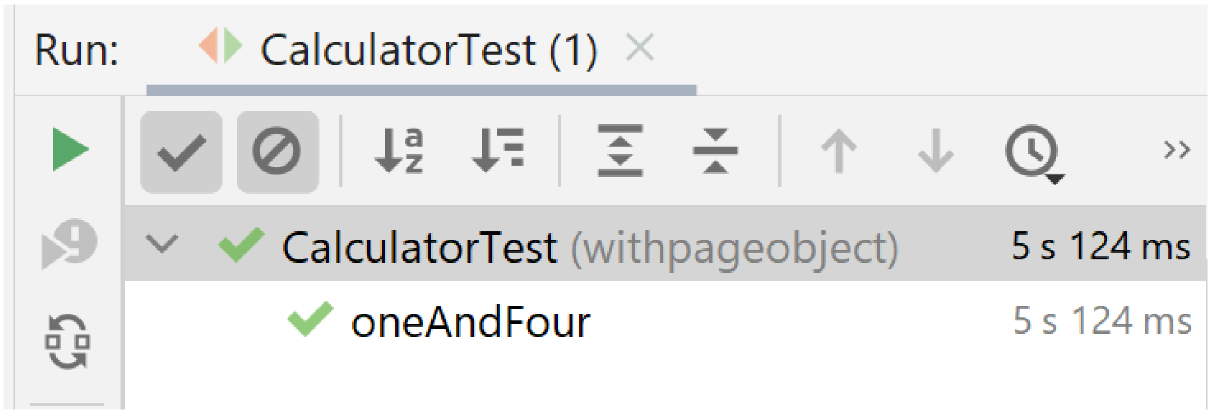
You can also right-click in the Elements tab, and select **Copy->Copy Selector** to get the CSS selector for this webelement.

You can also right-click in the Elements tab, and select **Copy->Copy Xpath** to get the XPath selector for this webelement.

Part 2:

In the same **WebdriverProject** you find an example using a page object in the folder **src/test/java/withpageobject**:

If you run the class **CalculatorTest** then this test succeeds without errors:



```
Run:  CalculatorTest (1)
✓ CalculatorTest (withpageobject)   5 s 124 ms
    ✓ oneAndFour                    5 s 124 ms
```

Write some new tests that tests the correct working of the calculator. To do this, you also have to add more code to the CalculatorPage class.

Part 3:

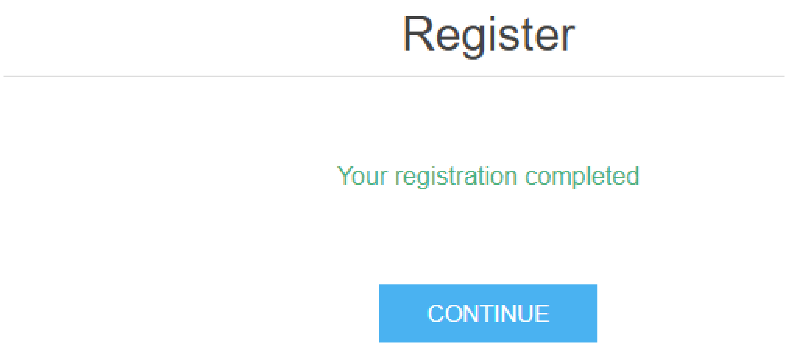
In React write a calculator that can add, subtract and multiply 2 numbers. Write a selenium test using a page object to test the calculator.

Part 4:

In React write a calculator that can add, subtract and multiply 2 numbers. When you enter the calculation information and click the add, subtract or multiply button, the application should navigate to the results page that shows the result of the calculation. Write a selenium test using a page object to test the calculator.

Part 5:

Write a new selenium webdriver tests for the registration process at the site
http://demo.nopcommerce.com/ using page objects
First click the registerbutton.
Then fill in the registration form
Then check if the new page shows the text "**Your registration completed**"



For this site you need to register everytime with an unique email address.
In java you can create a unique email address using a random number:

```
private String createUniqueEmail() {
      String email="@gmail.com";
      String name="frank"+ createRandomNumber();
      return name+email;
}

private int createRandomNumber() {
      return (int) (Math.random() * 5000 + 1);
}
```

Part 6:

Write a new selenium webdriver tests that test the correct working of the application you wrote for lab10 part 2.

What to hand in?

A separate zip file for each part.