

# Dia 3: Tests d'interfície d'usuari

Xavier Sala Pujolar



Universitat  
de Girona

Febrer 2021

# Tests end-2-end



Es comprova que el programa complet funciona des del punt de vista de l'usuari

Es fan proves que usen tota l'aplicació

Es fan servir components reals, rarament es fan servir dobles

És habitual tenir  
tests que posen a  
prova la UI

Fer servir  
els controls

Entrar  
dades



# Eines per end-to-end tests

---

3 Fases

Given

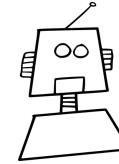
When

Then

Preparar l'entorn per fer  
els resultats previsibles

Màquines  
virtuals,  
contenidors

Simulació de  
l'usuari

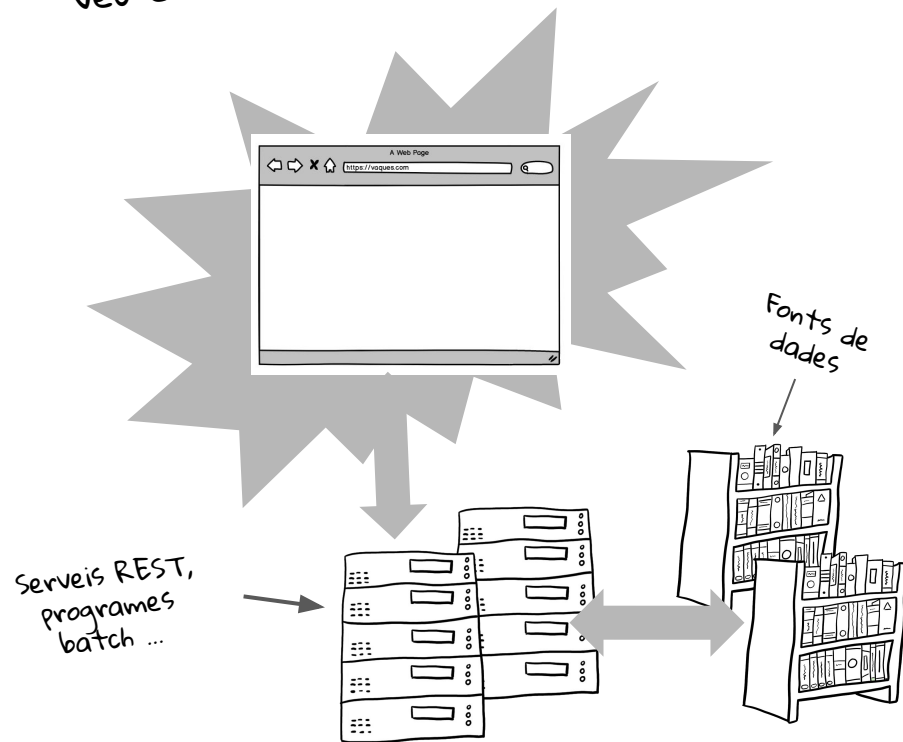


Eines de  
comprovació

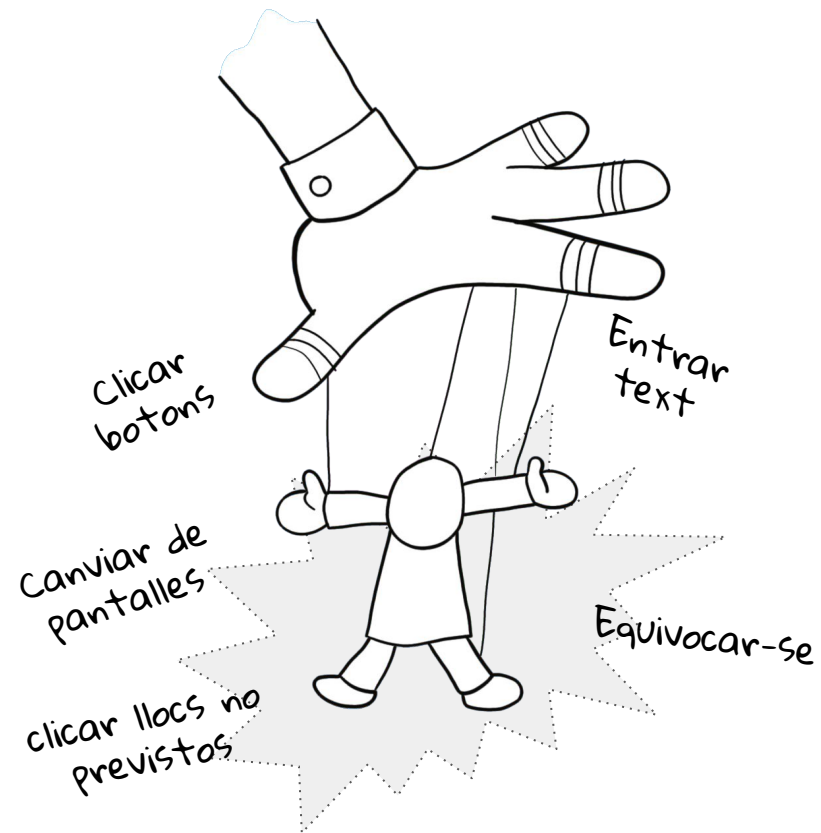


Es fan sobre  
l'aplicació en marxa

El test es fa sobre el que es  
veu en la interfície d'usuari



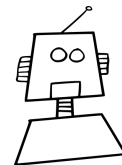
Es simulen les  
interaccions dels  
usuaris



**Selenium** són un grup  
d'eines de **codi obert**  
d'automatització de  
navegadors

Es controla el  
navegador des del  
test

Java, C#,  
Python,  
Javascript, PHP,  
Ruby, Perl



Es poden anar  
fent  
comprovacions  
sobre el codi  
HTML que hi  
surts

Comprovar l'HTML

Buscar-hi amb

- Expressions CSS
- XPath
- Etiquetes
- ....

```
<html>
<head>
  <title>A Web
  <css> ...
</head>
<body>
  <div id="mai
  </div>
</body>
</html>
```

Es poden emplenar  
camps de text

Fer servir la  
interfície

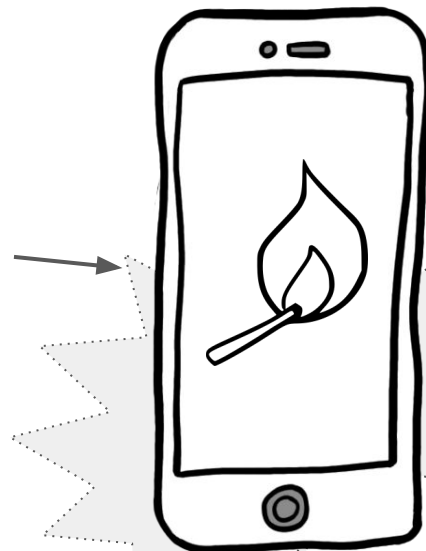
Prémer botons

Moure  
l'scroll



**Espresso** és una llibreria  
d'automatització per  
aplicacions Android

El test  
controla el  
mòbil



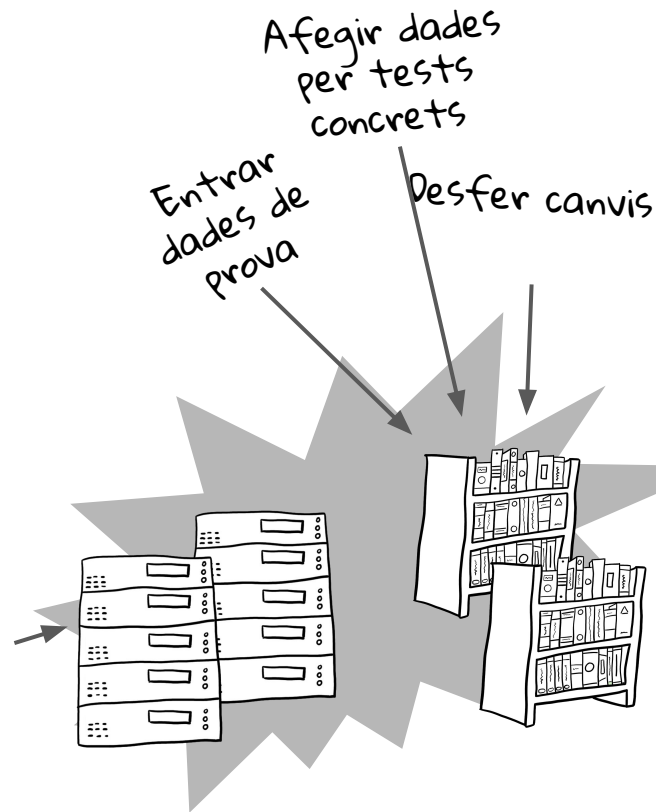
Es pot  
comprovar què  
li passa a la  
UI



Els tests han de ser  
**repetibles** i per tant  
són lents

Per això no es  
solen executar  
sempre

Iniciar  
l'aplicació i tot  
el que li fa  
falta



És fàcil tenir **falsos  
negatius**



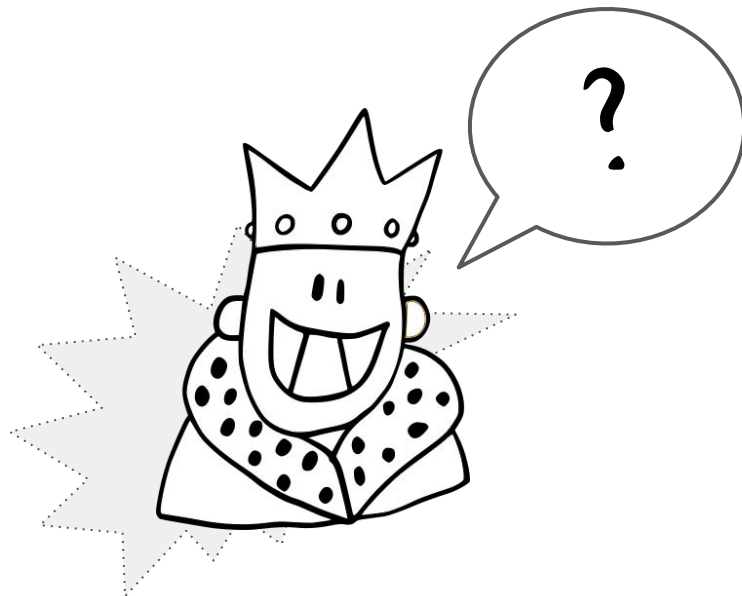
# Llegibilitat dels tests

A vegades es fan servir els tests per comprovar els requeriments.

Els tests es poden fer servir per demostrar als clients que s'estan complint els requeriments

# Millorar la llegibilitat dels tests

A vegades cal que  
usuaris **no**  
**informàtics** donin la  
seva aprovació




Els sistemes basats  
en **Gherkin** defineixen  
tests en text planer



**Feature:** Afegir gent a una Casa  
Funcionament del mecanisme d'afegir  
gent a una Casa

**Scenario:** Afegir una persona a una casa  
**Given** Una casa amb 3 persones  
**And** Una persona que està fora  
**When** La persona entra a la casa  
**Then** Hi ha d'haver una persona més


Es defineixen les  
frases de cada  
fase



```
[Given(@"Una casa amb (.*) persones")]  
public void GivenUnaCasa(int gent)  
{  
    casa = new Casa(gent);  
}
```

**Feature:** Afegir gent a una Casa  
Funcionament del mecanisme d'afegir  
gent a una Casa

**Scenario:** Afegir una persona a una casa  
**Given** Una casa amb 3 persones  
**And** Una persona que està fora  
**When** La persona entra a la casa  
**Then** Hi ha d'haver una persona més



Ja n'hi ha  
prou de  
rotllo!



P1: Per fi anem per feina



# Provar el funcionament de la previsió del temps

```
$ dotnet run
```





- Comprova que no es veu res sense fer login
- Comprova el funcionament del login
- Comprova que es pot registrar un usuari
- Comprova que un usuari correcte pot veure les previsions
- Comprova que les previsions d'un poble són les mateixes encara que hi vagis dos cops
- ...

# Provar Adopta una vaca

```
$ docker run -p 8080:80 -d utrescu/adoptavaca
```

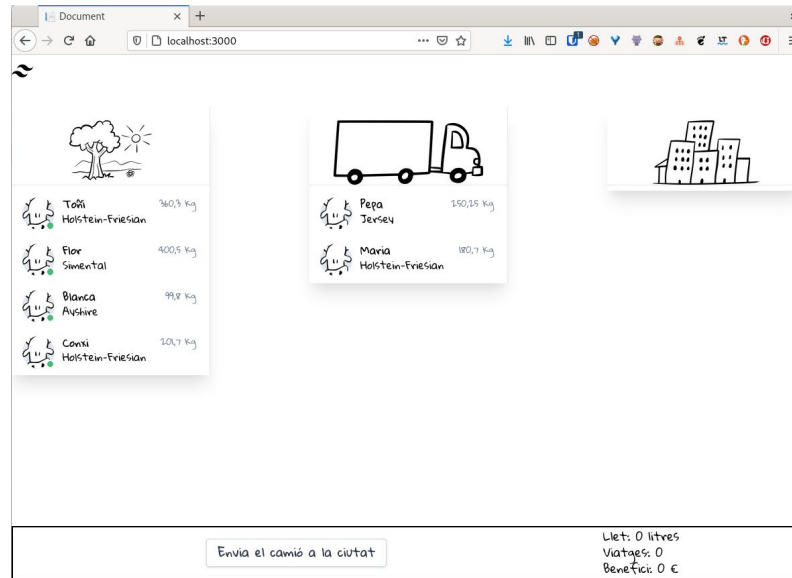


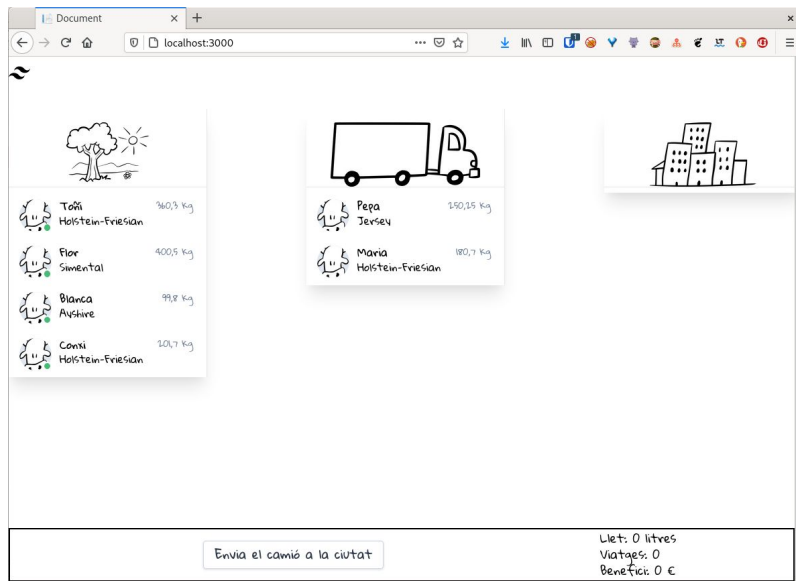


- Comprova que els usuaris amb contrasenyes incorrectes no poden entrar
- Comprova que no s'entra si no hi ha usuari o contrasenya
- Comprova que un usuari correcte pot entrar:
  - *matilda : adlitam*
  - *conxita : energy*
  - *peluda : password*
- Comprova que un usuari que ha entrat si surt no pot tornar a la pàgina privada.
- ...

# Provar el funcionament del transport de vaques

```
$ docker-compose up
```





- Comprova que es poden posar vaques al camió
- Comprova que no es poden posar més de 1000 kg en el camió
- Comprova que es poden treure vaques del camió
- Comprova que es poden enviar les vaques a la ciutat
- Comprova que els valors de la barra canvien
- ...

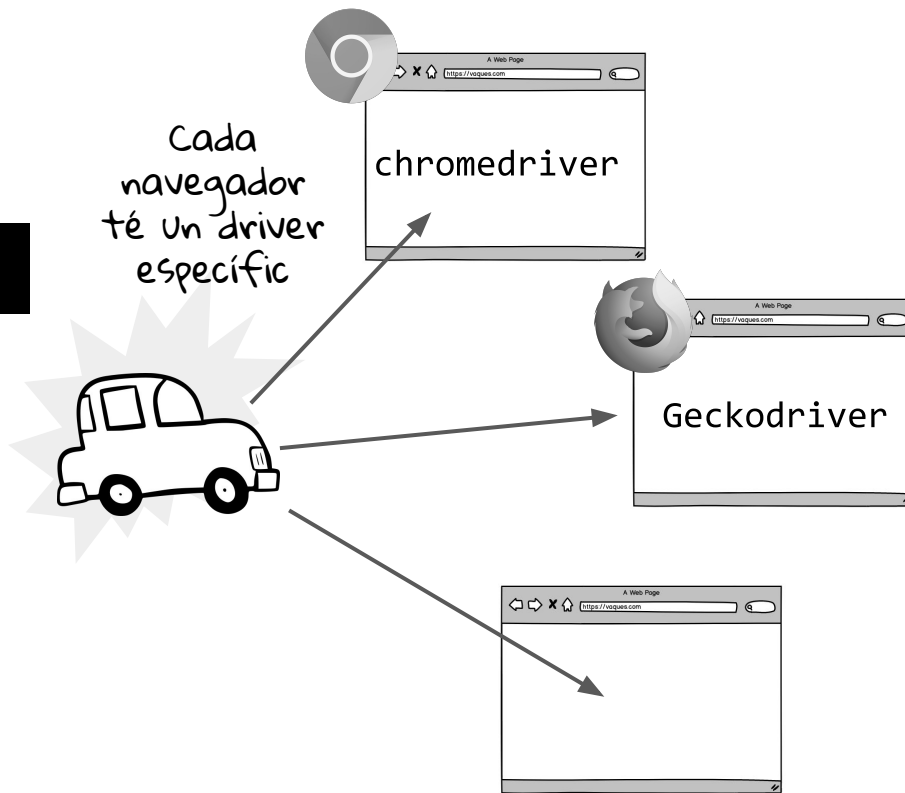
Resum de les  
eines que usarem  
en les pràctiques





Es crea un  
Webdriver que  
controlarà els  
navegadors

```
var driver = new ChromeDriver();
```





```
driver.Navigate().To("http://www.vaques.com")
```

```
driver.Navigate().Refresh()
```

```
driver.Navigate().Back()
```

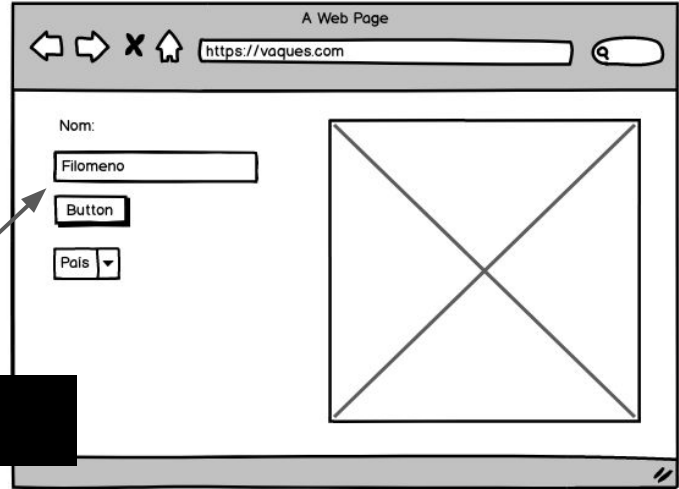
```
driver.Navigate().Forward()
```

Amb el driver es  
pot controlar el  
navegador

```
driver.Manage().Window().Maximize()
```



Localitzar elements  
de la pàgina



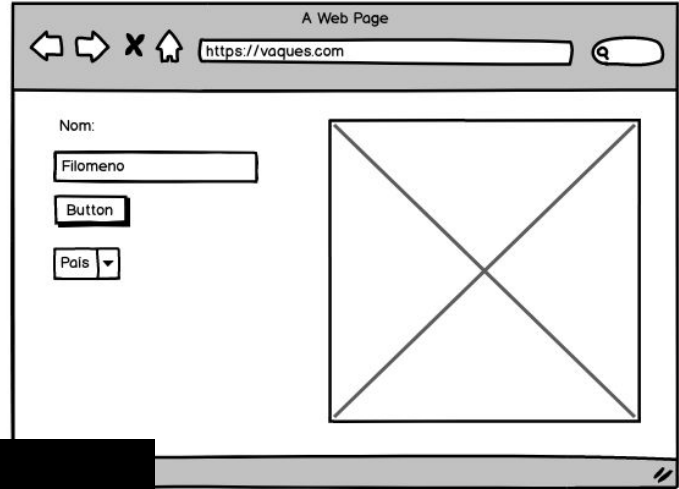
```
var input = driver.FindElement(By.Id("nom"));
```

```
By.CssSelector(".. By.LinkText("..  
By.XPath("...") By.PartialLinkText("...")  
By.ClassName("...' By.TagName("...")
```

```
FindElements(By.Id("nom"));
```

Més d'un  
resultat?

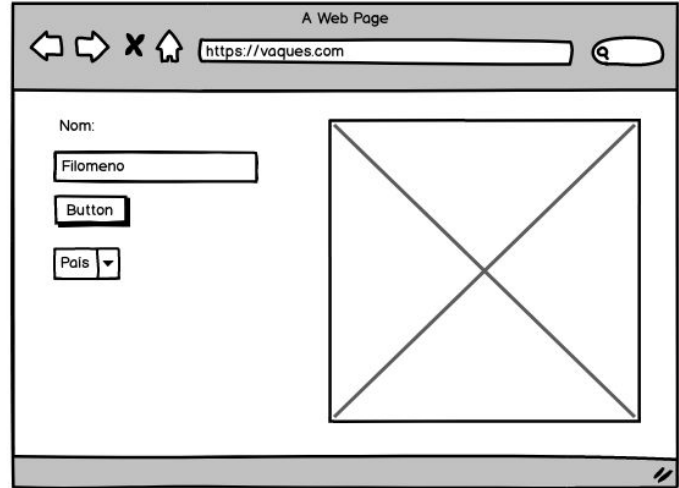
Interactuar amb els  
elements



```
var input = driver.FindElement(By.Id("nom"));  
input.SendKeys("Filomeno");
```

```
driver.FindElement(By.Class("submit")).Click();
```

Interactuar amb els  
elements

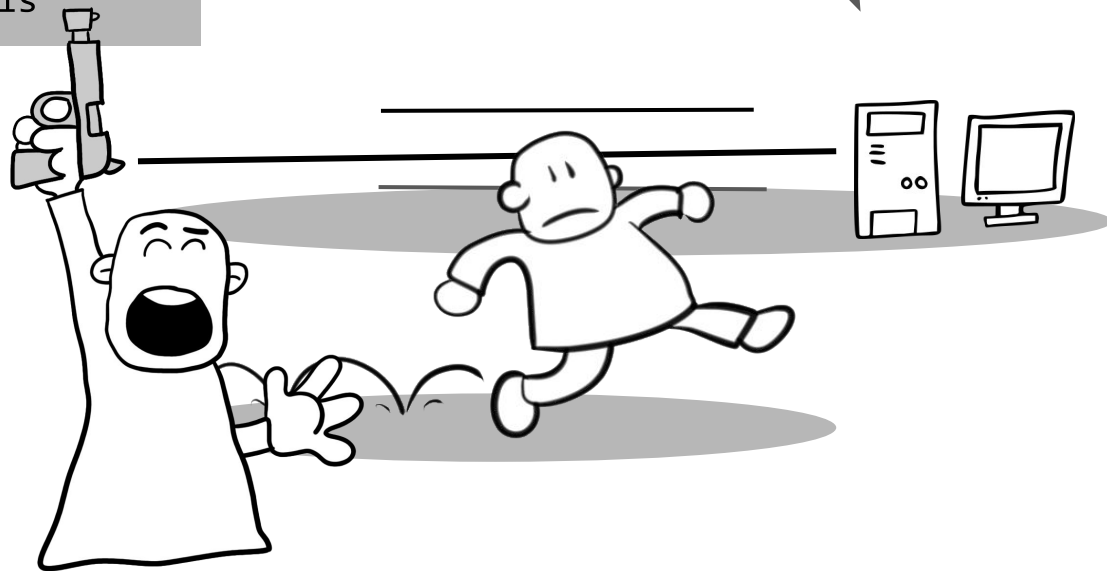


```
var text = driver.FindElement(By.Id("boto")).Text;
```

```
var text = driver.FindElement(By.Id("boto"))  
    .GetAttribute("type");
```

Normalment els  
programes són més  
ràpids que els  
usuaris

Això fa que a vegades els  
components encara no  
hagin aparegut



Es poden definir  
esperes

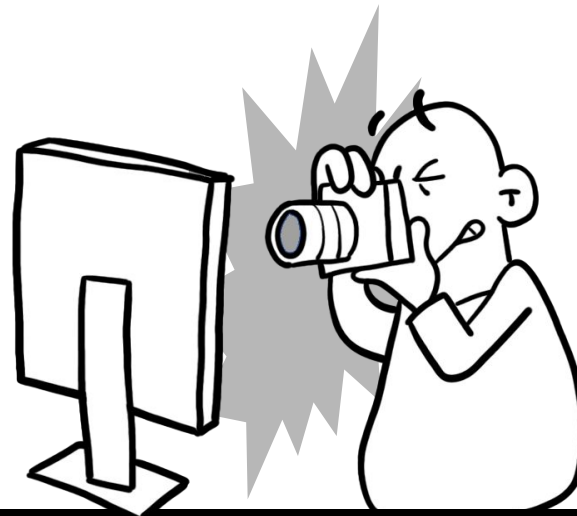


Com a màxim  
espera 10  
segons

```
WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));  
wait.PollingInterval = TimeSpan.FromMilliseconds(100);
```

Ho mira  
cada 100  
mil·lisegons

Té altres  
possibilitats com fer  
captures de pantalla,  
...



```
Screenshot image = ((ITakesScreenshot)driver).GetScreenshot();  
image.SaveAsFile("C:/temp/Screenshot.png", ImageFormat.Png);
```

