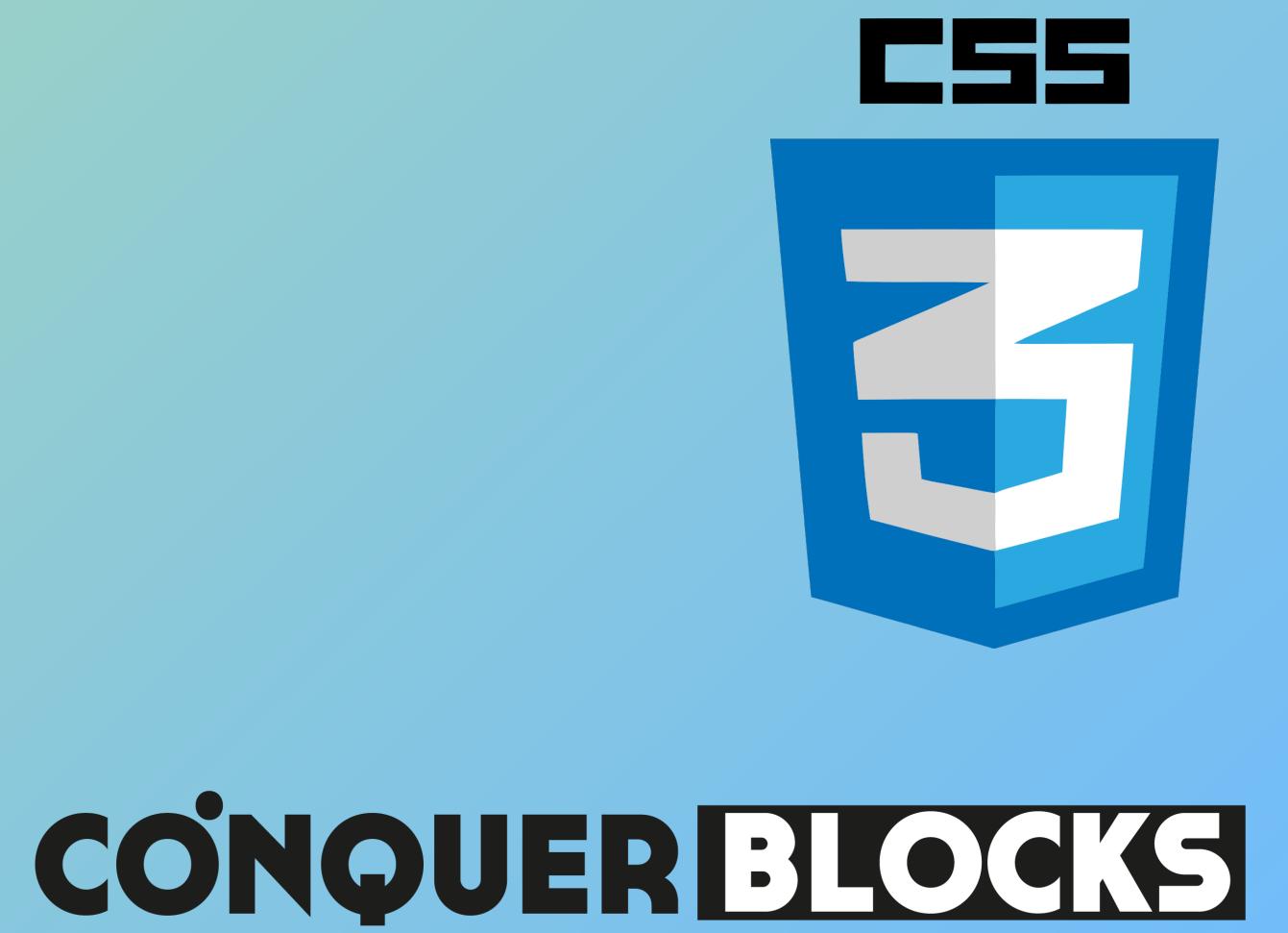


# {css}

Clase 31



# Clase 31

# <índice>

## Responsive Web Design

¿Qué es?

---

Viewport

---

Instrucción @media

---

Breackpoints típicos

---

Mobile First vs Desktop First

---

Min/Max Height/Width Min-content/Max-content

---

# ¿Qué es RWD?

**Ya no es una tendencia**

Más del **60% del tráfico en internet** se hace  
desde un dispositivo móvil

Es **obligatorio** aplicarlo a nuestros proyectos

El diseño web adaptable, o responsive web design, es una metodología de diseño y desarrollo cuyo objetivo es crear sitios web que proporcionen una experiencia de visualización óptima en un amplio rango de dispositivos, desde computadoras de escritorio hasta teléfonos móviles y tabletas.

Esto implica que el **diseño** debe ser capaz de ajustarse fluidamente a diferentes tamaños y resoluciones de pantalla, y a diferentes entornos, garantizando legibilidad y usabilidad sin importar el dispositivo.

¿Cómo funcionaban antes del RWD los sitios web?

¿Cómo funcionaban antes del RWD los sitios web?

¿Por qué tu web debe ser responsive?

## Beneficios

Mejor experiencia de usuario

Se acabaron los contenidos duplicado => SEO

Reducción de costes de desarrollo

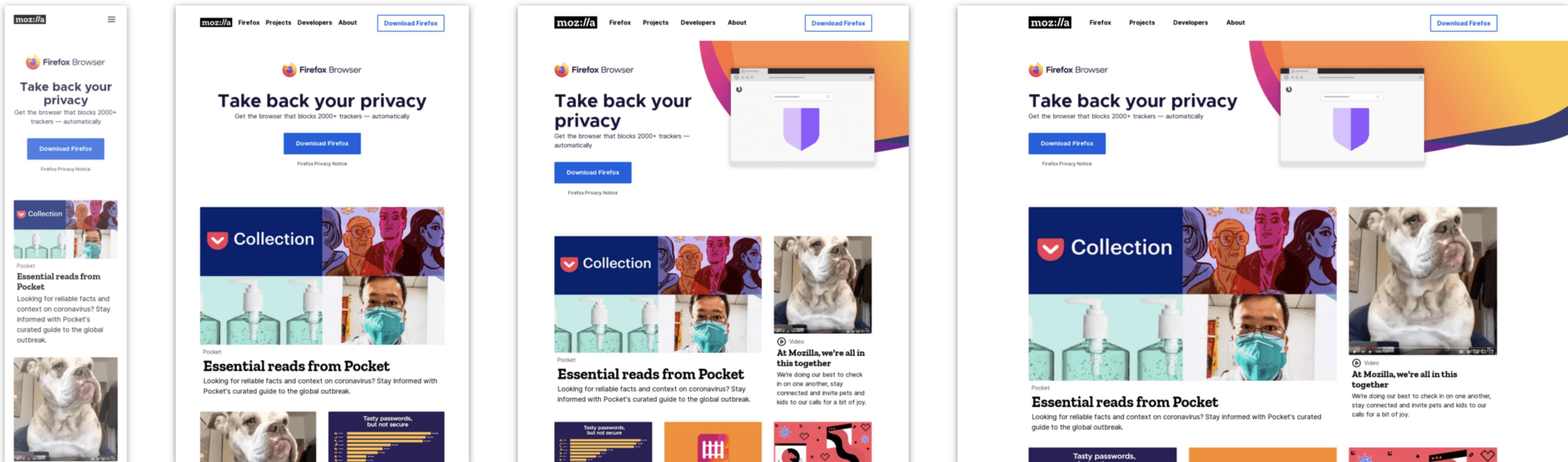
Aumenta el alcance móvil de tus contenidos

Soporte para futuros dispositivos

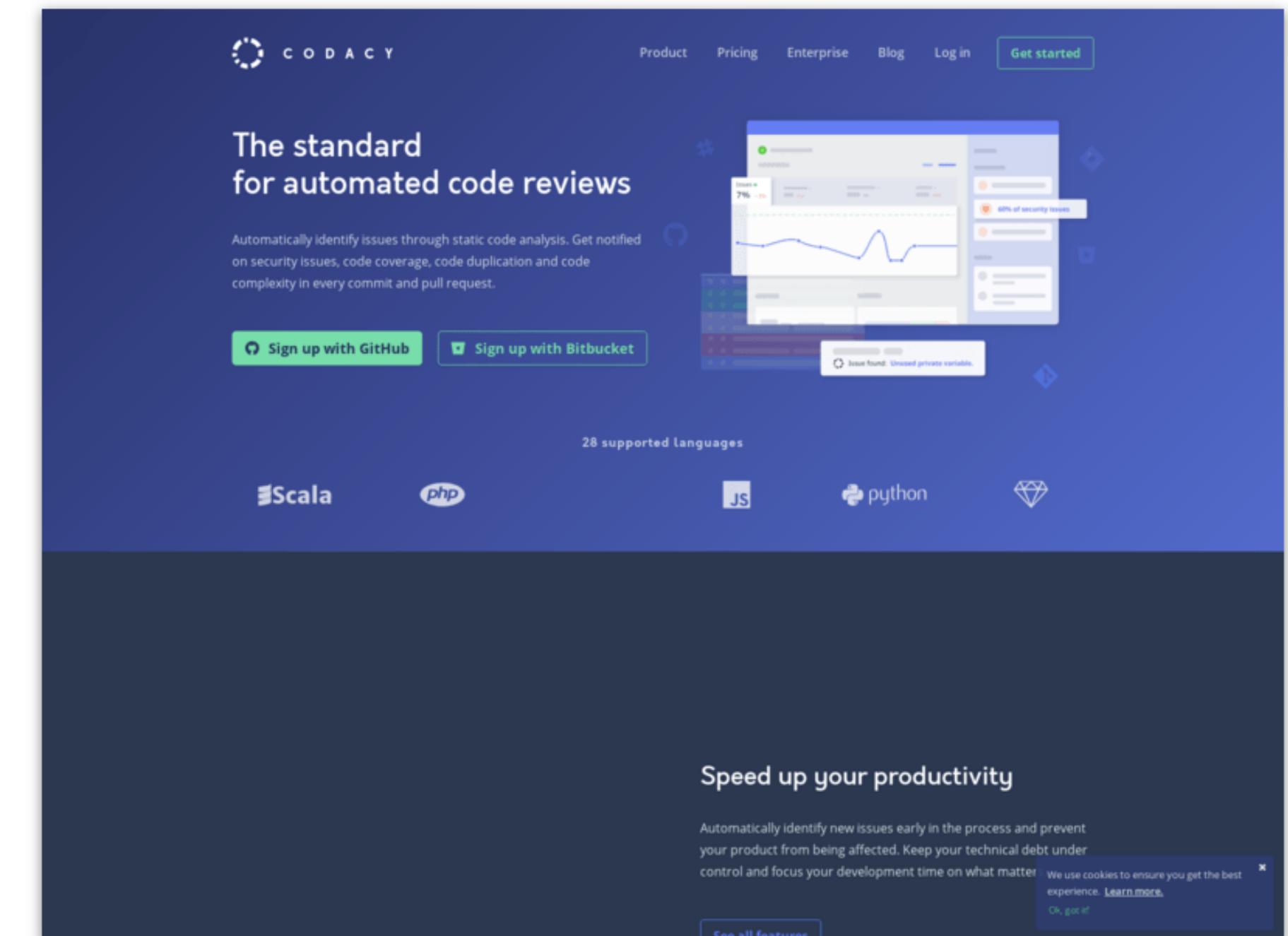
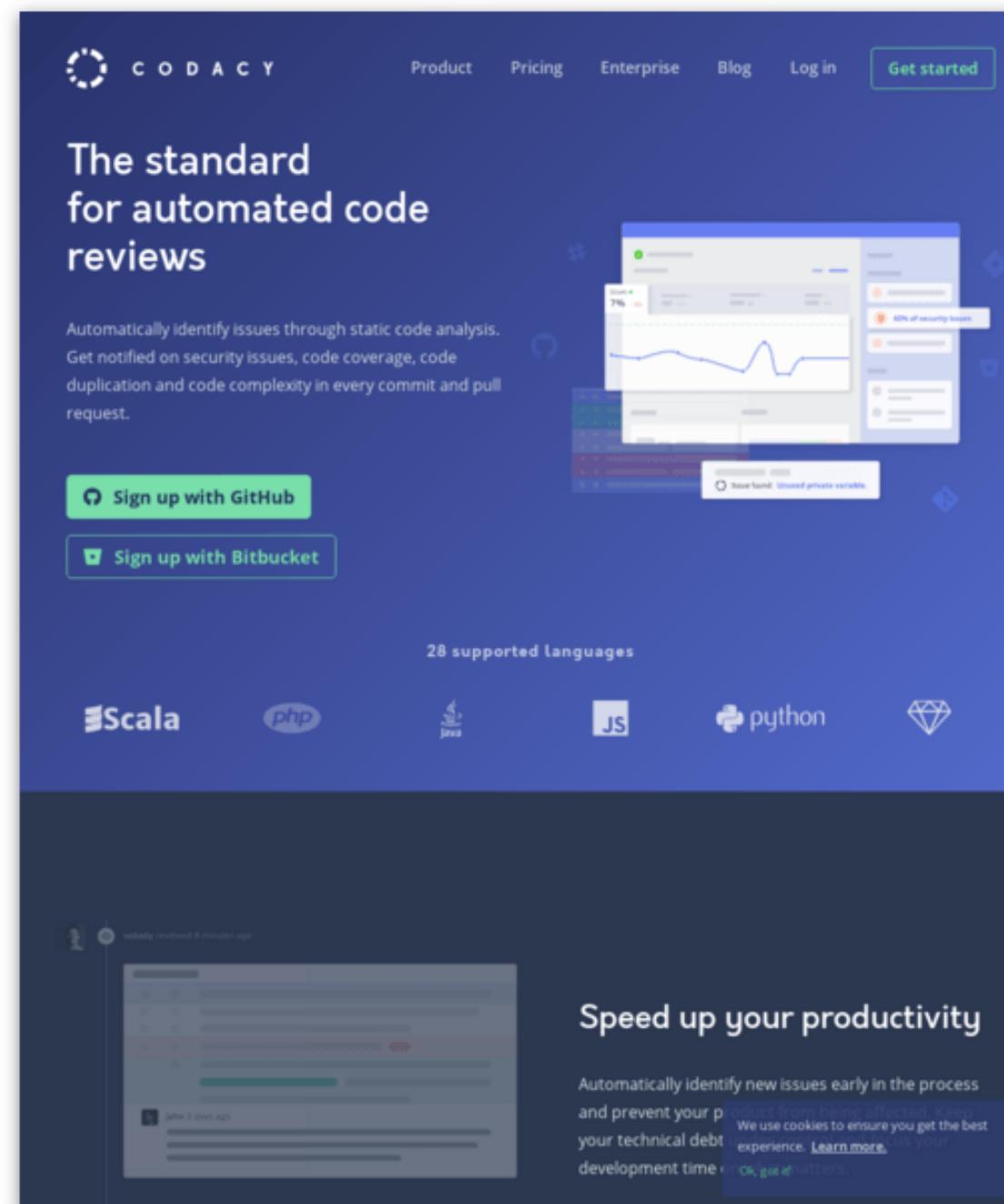
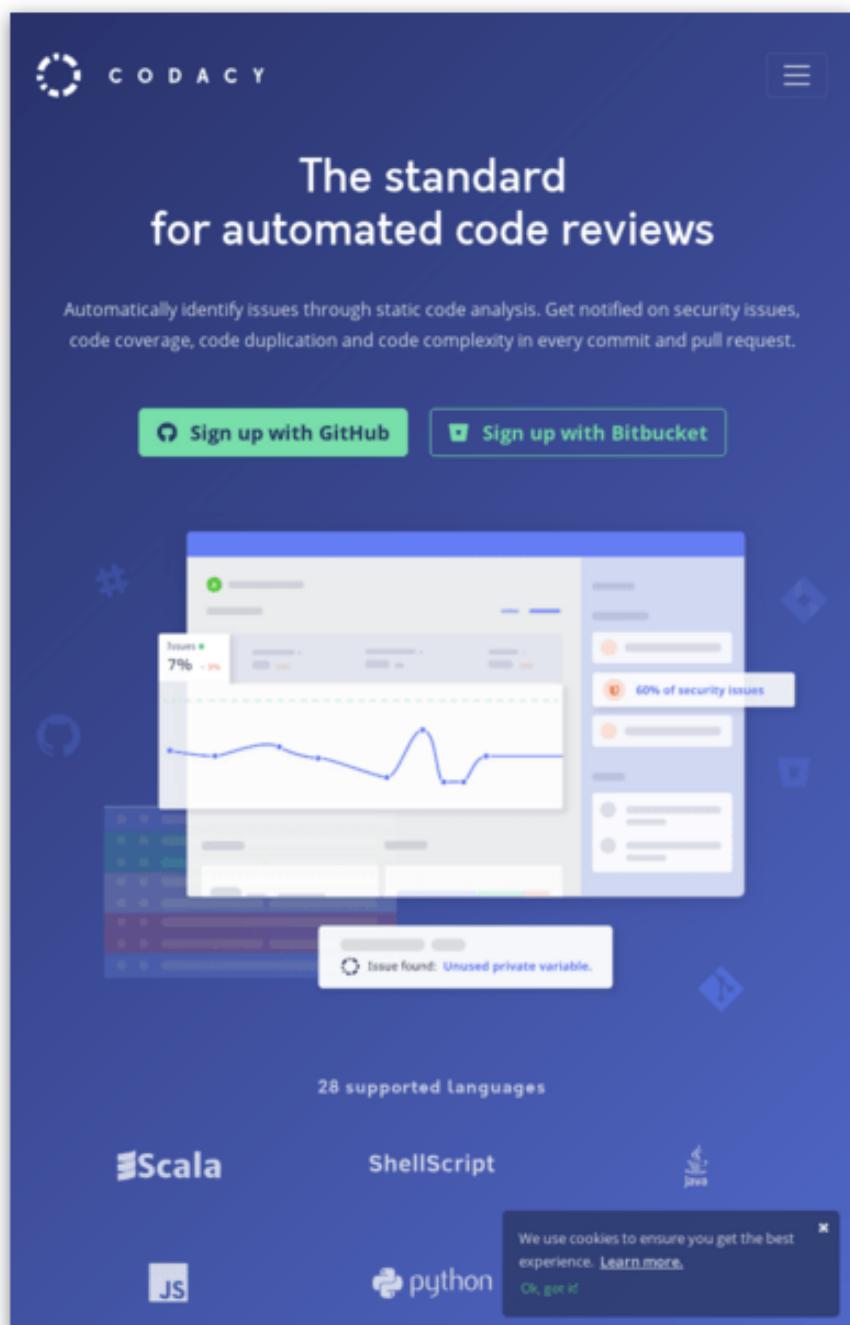
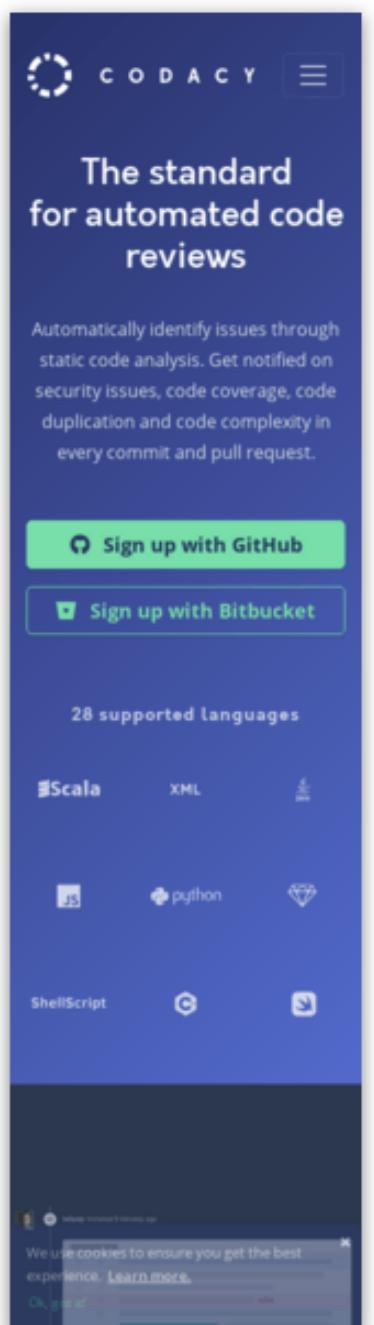
Mantenimiento mucho más sencillo

## Mozilla

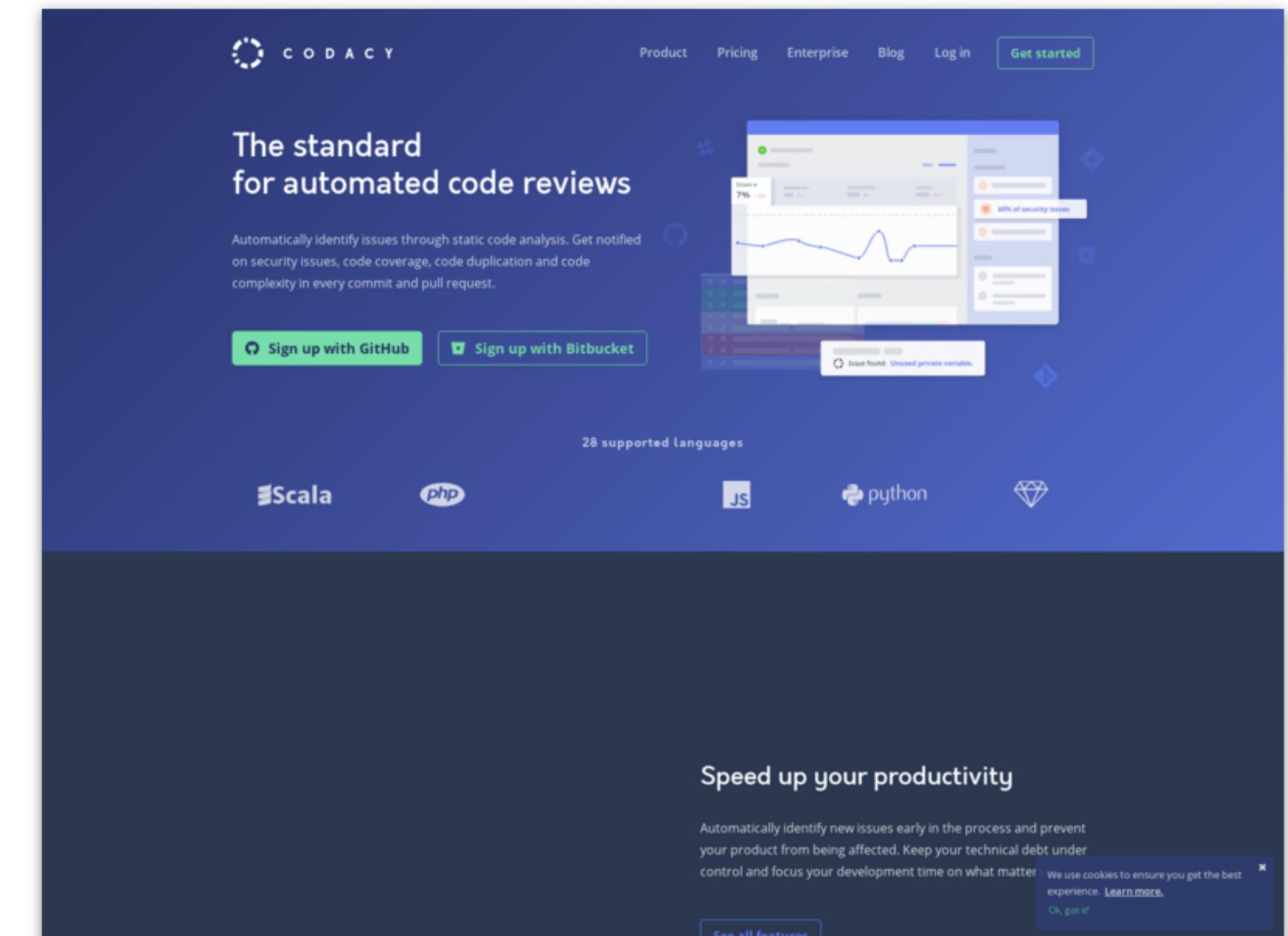
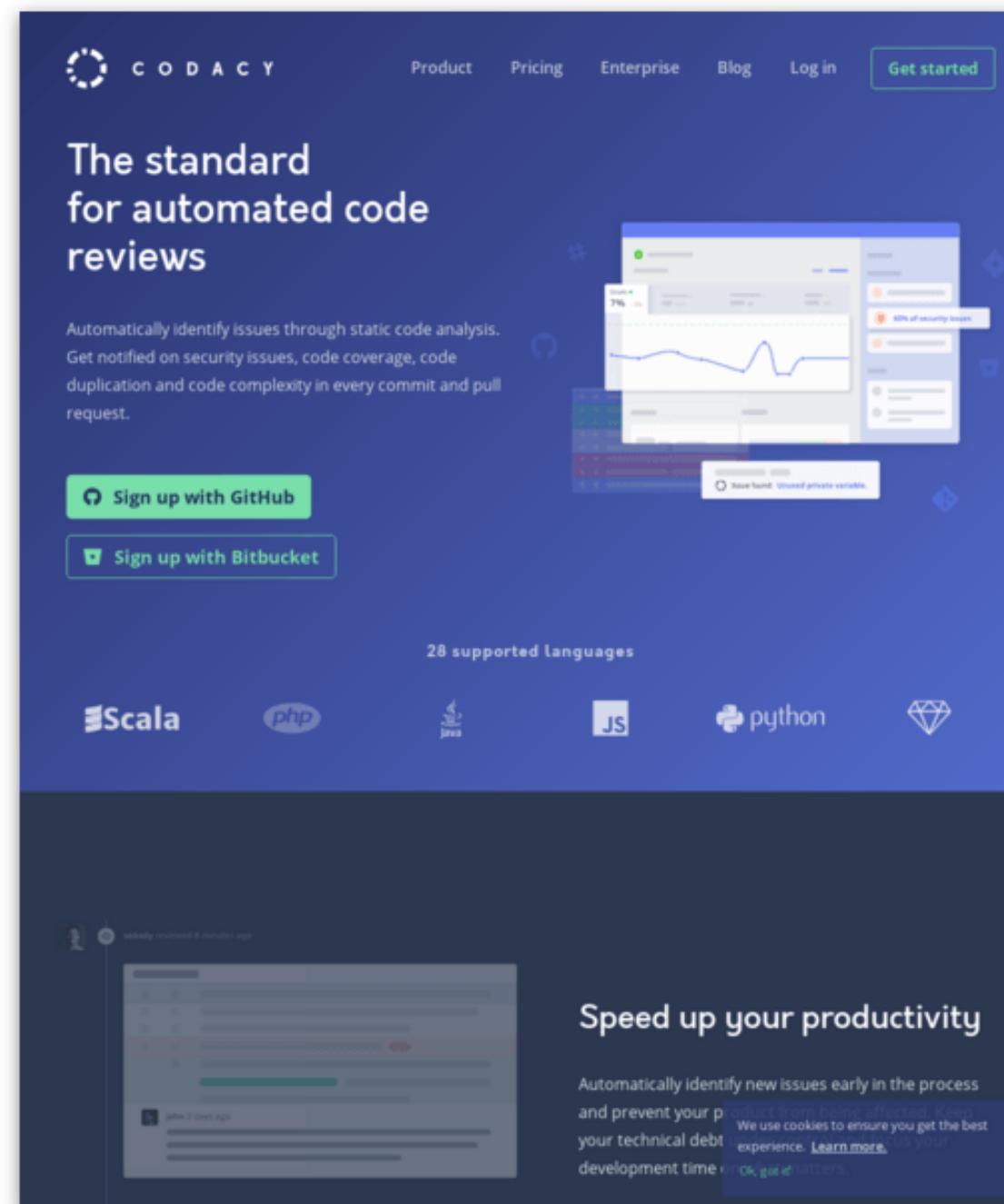
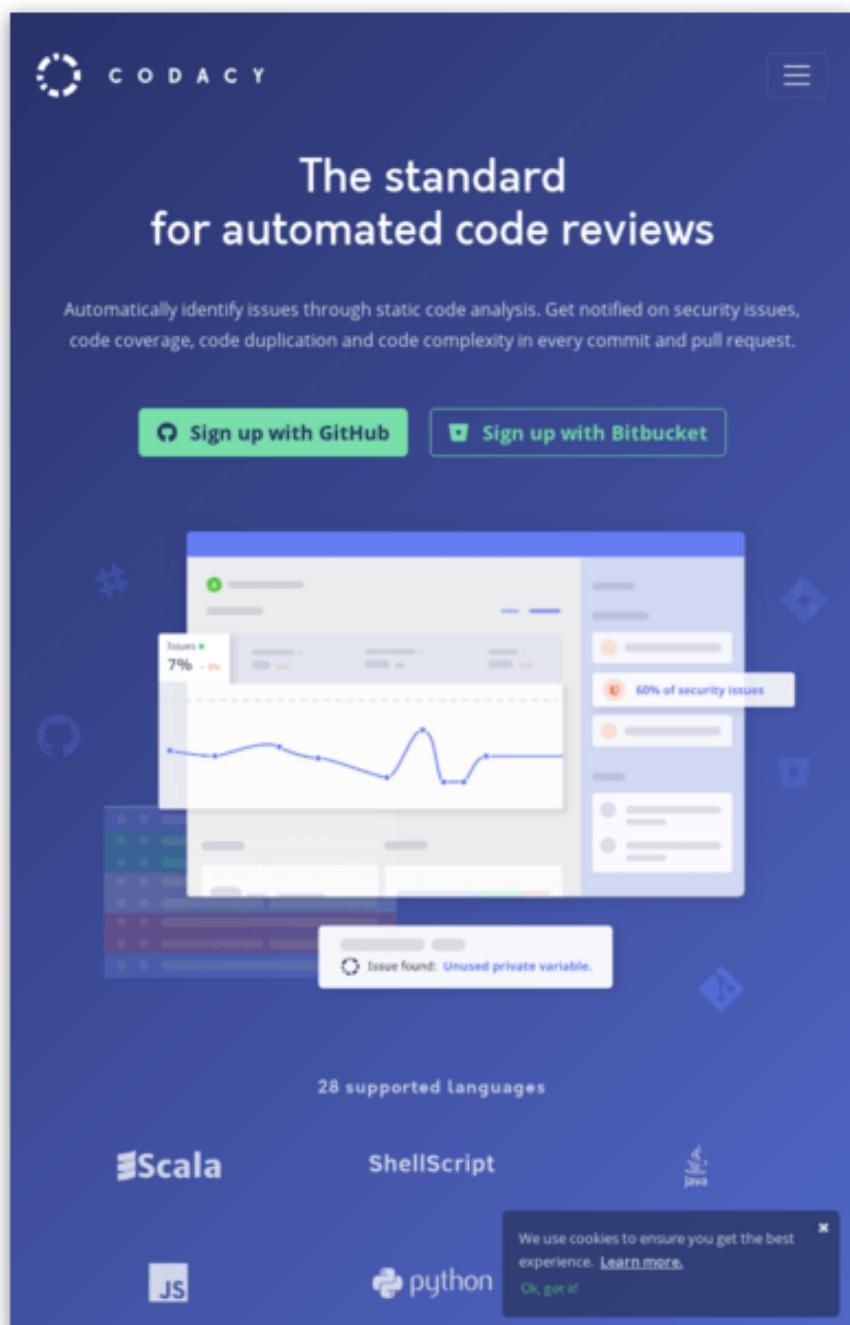
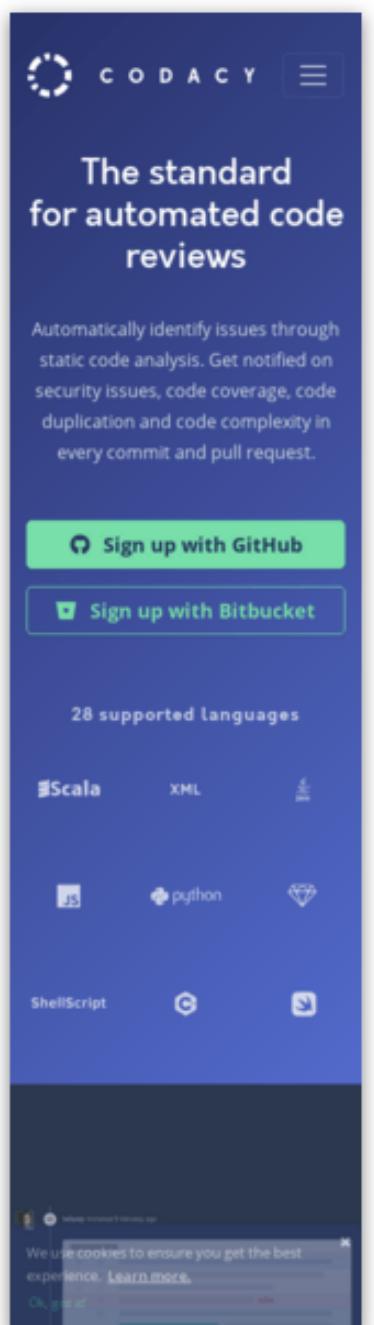
9



## Codacy



## Codacy



## PullRequest

5

The image displays four wireframe prototypes of a code review platform's landing page, arranged horizontally. Each prototype shows a different service offering:

- Code review as a service for MySQL|**: Shows a person and a robot at a desk.
- Code review as a service for any |**: Shows a person and a robot at a desk.
- Code review as a service for MySQL|**: Shows a person and a robot at a desk.
- We review within your tools.**: Shows various developer tools like Docker, GitHub, and Xcode.

Each prototype includes sections for "Sign Up" and "Book a Chat". The bottom section of the fourth prototype features a screenshot of a code editor with a pull request message from a bot, followed by a dark blue footer bar.

# Principios básicos

# Principios básicos

El primero de ellos es la diferencia entre diseño responsive y diseño adaptativo. Como se puede ver en la imagen a continuación, un diseño responsive responde (valga la rebuznancia) en todo momento a las dimensiones del dispositivo, mientras que un diseño adaptable es aquel que se adapta, pero no necesariamente responde en todo momento

# Principios básicos

## Responsive vs Adaptive

Responsive

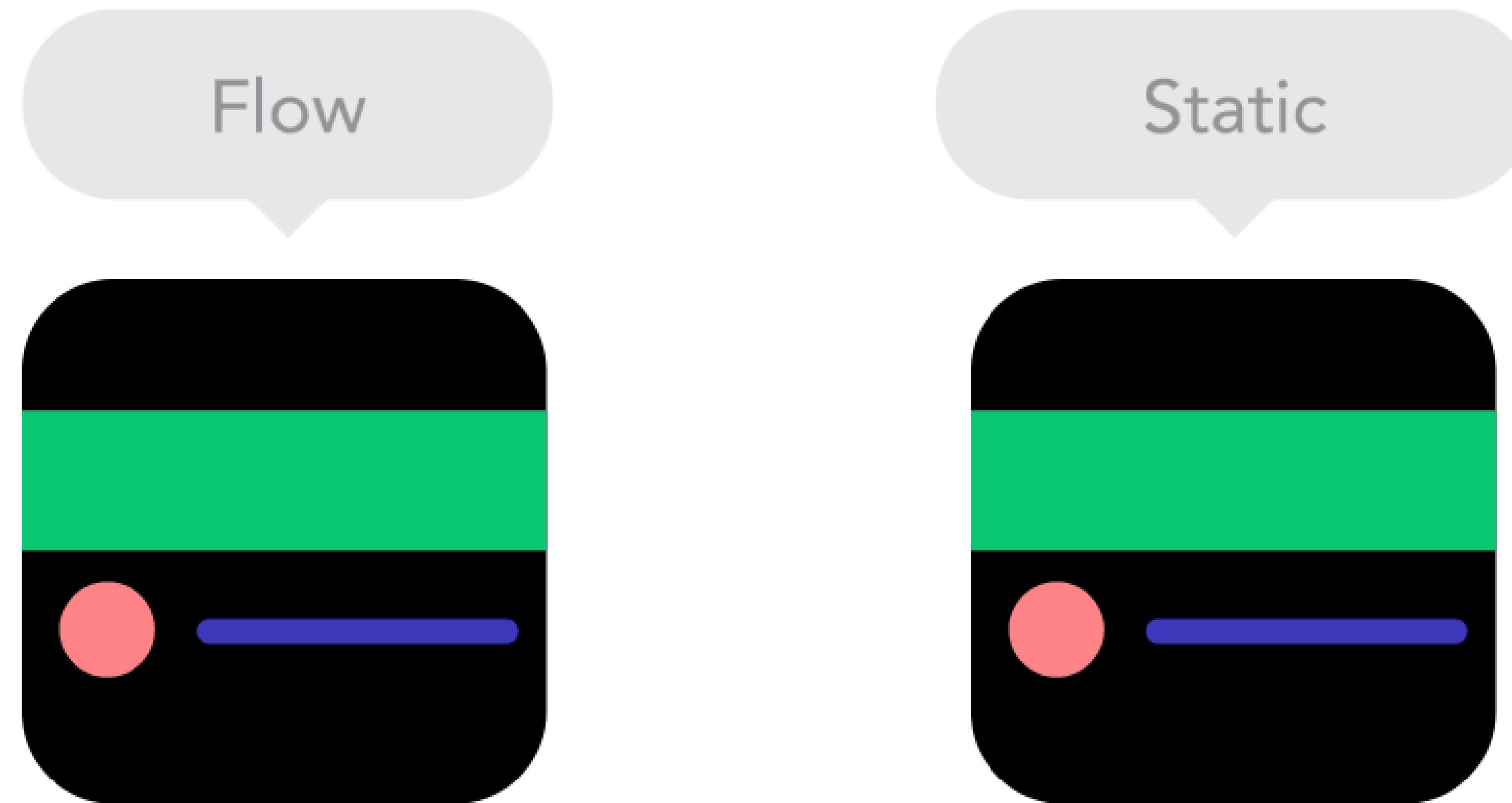


Adaptive



# Principios básicos

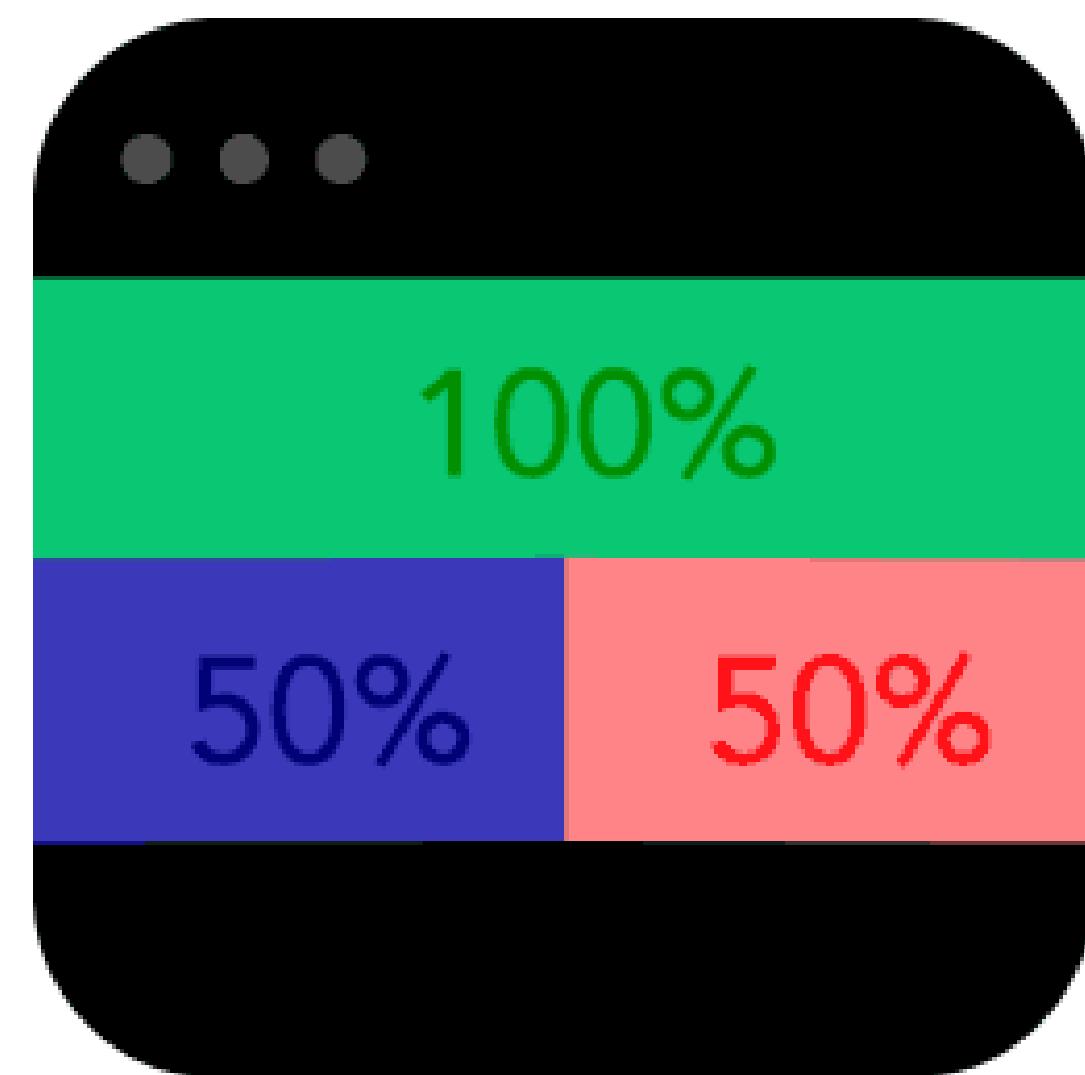
No cortemos el flow normal de elementos



# Principios básicos

Usemos medidas relativas siempre que podamos

Relative Units



Static Units



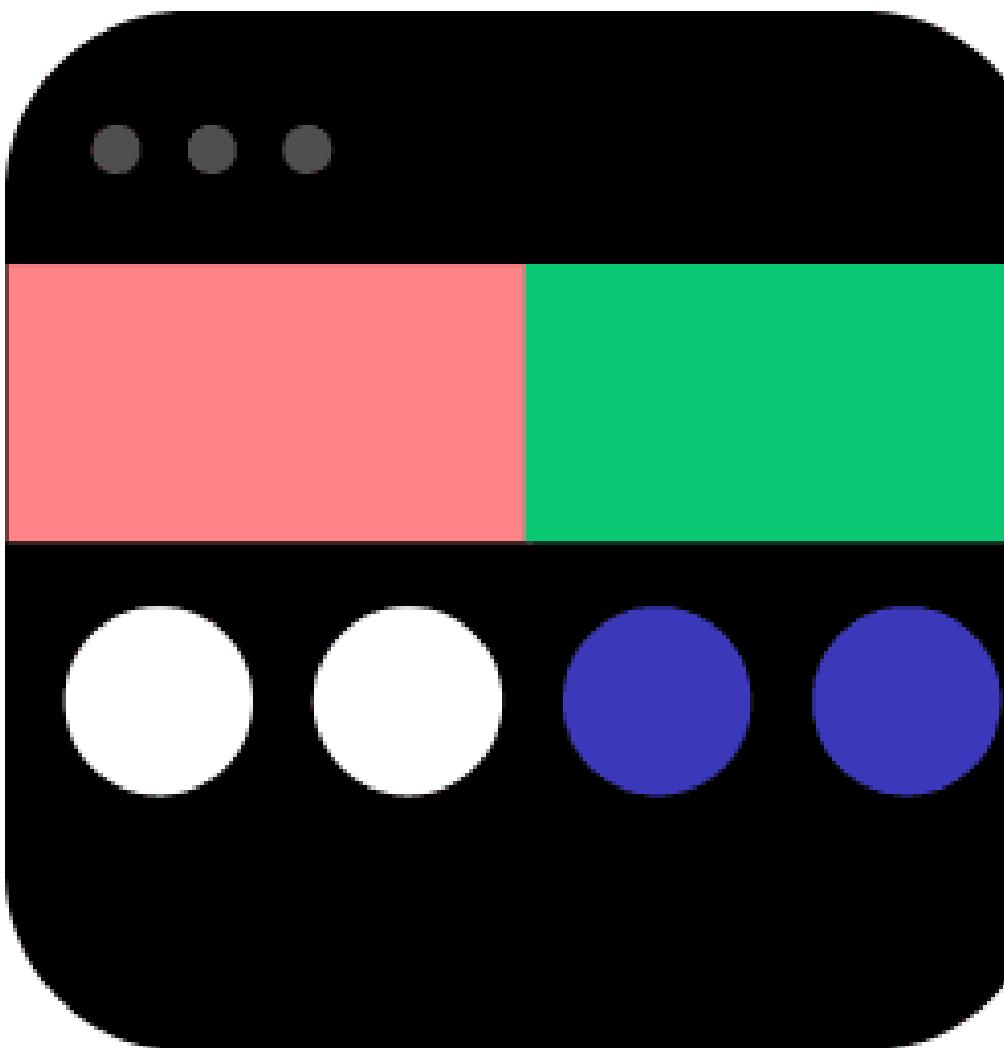
# Principios básicos

Usaremos breackpoints

With Breakpoints



Without Breakpoints



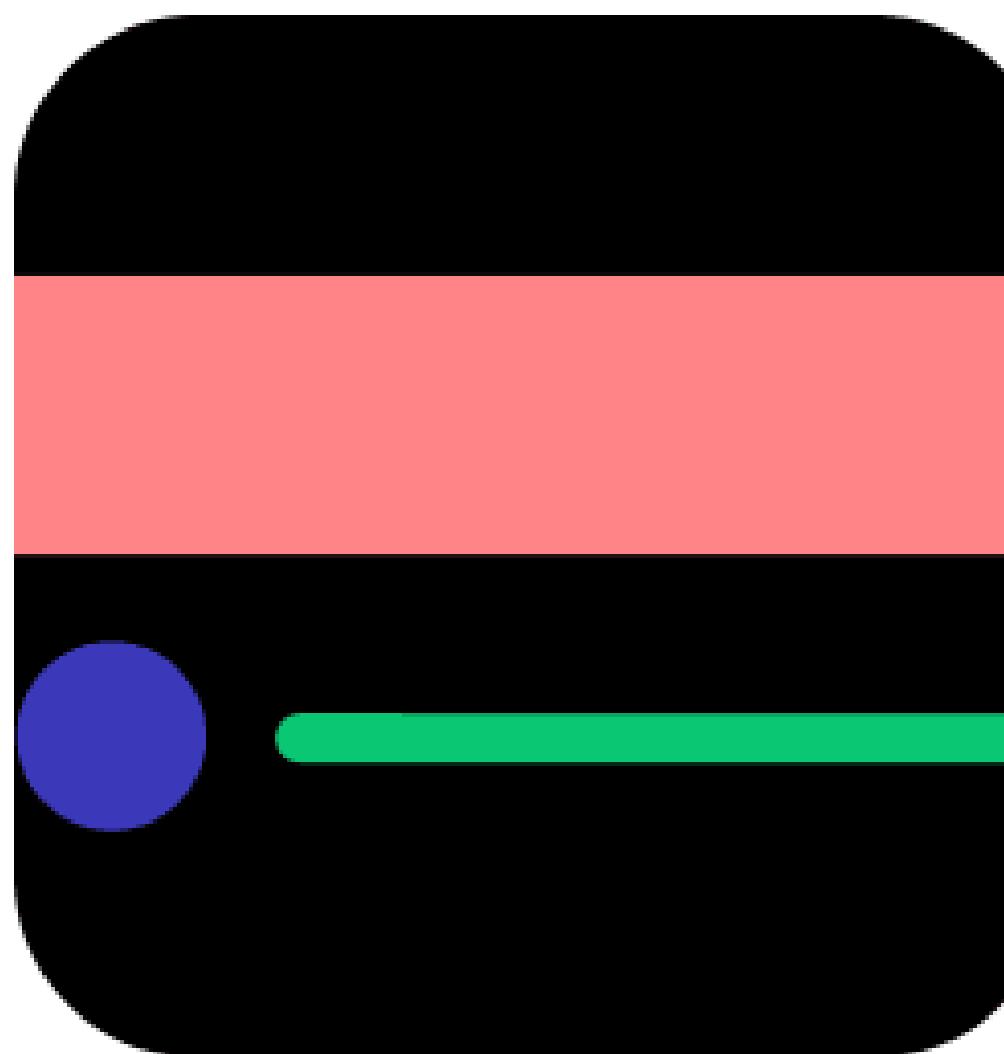
# Principios básicos

## Controlemos anchos máximos

Max width



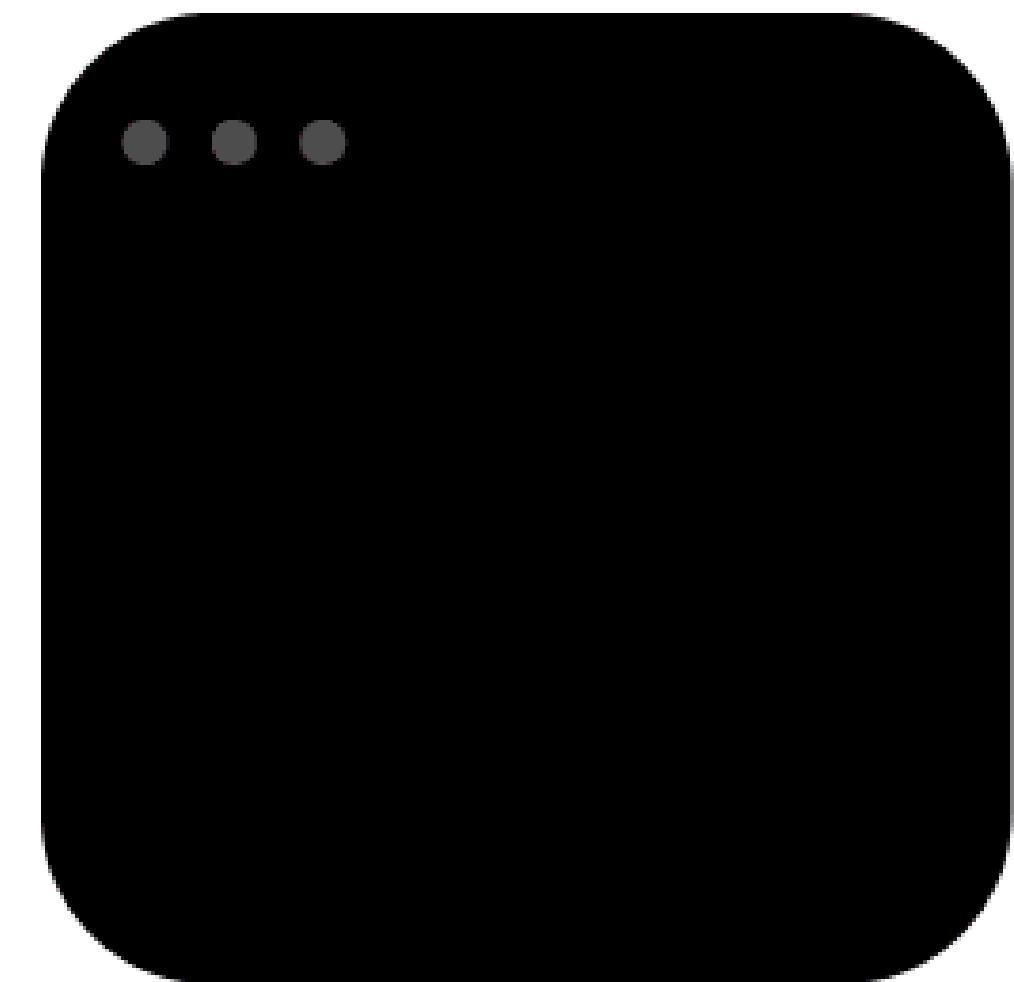
No max width



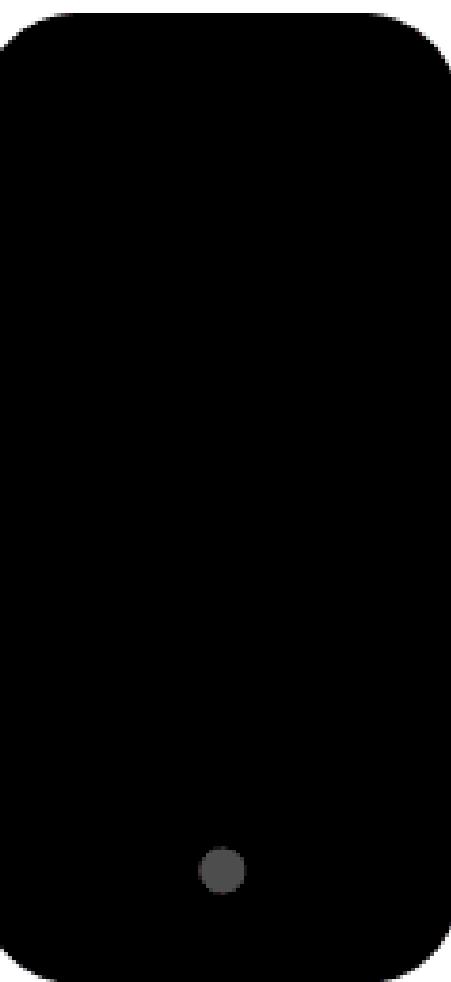
# Principios básicos

Usemos una metodología apropiada

Desktop first



Mobile first



# Principios básicos

Usemos webfonts en lugar de system fonts

System fonts



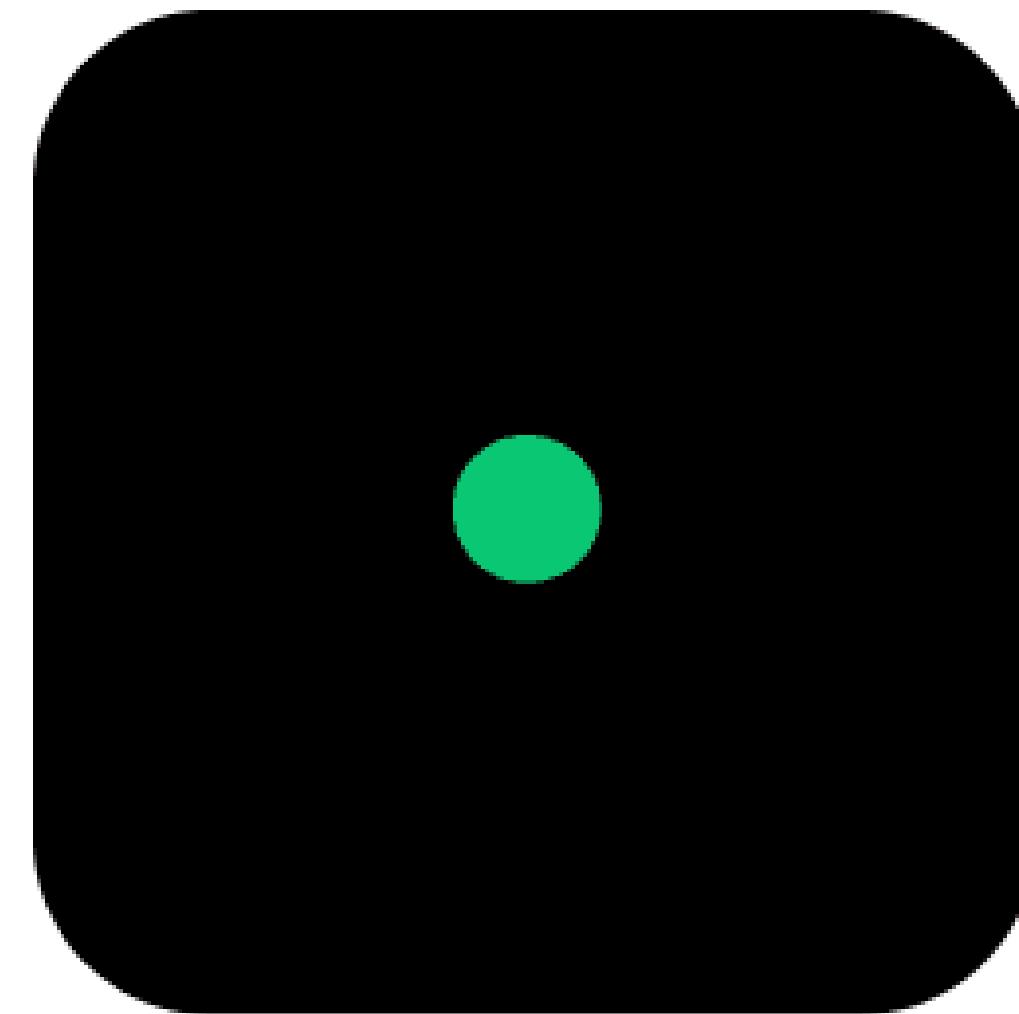
Webfonts



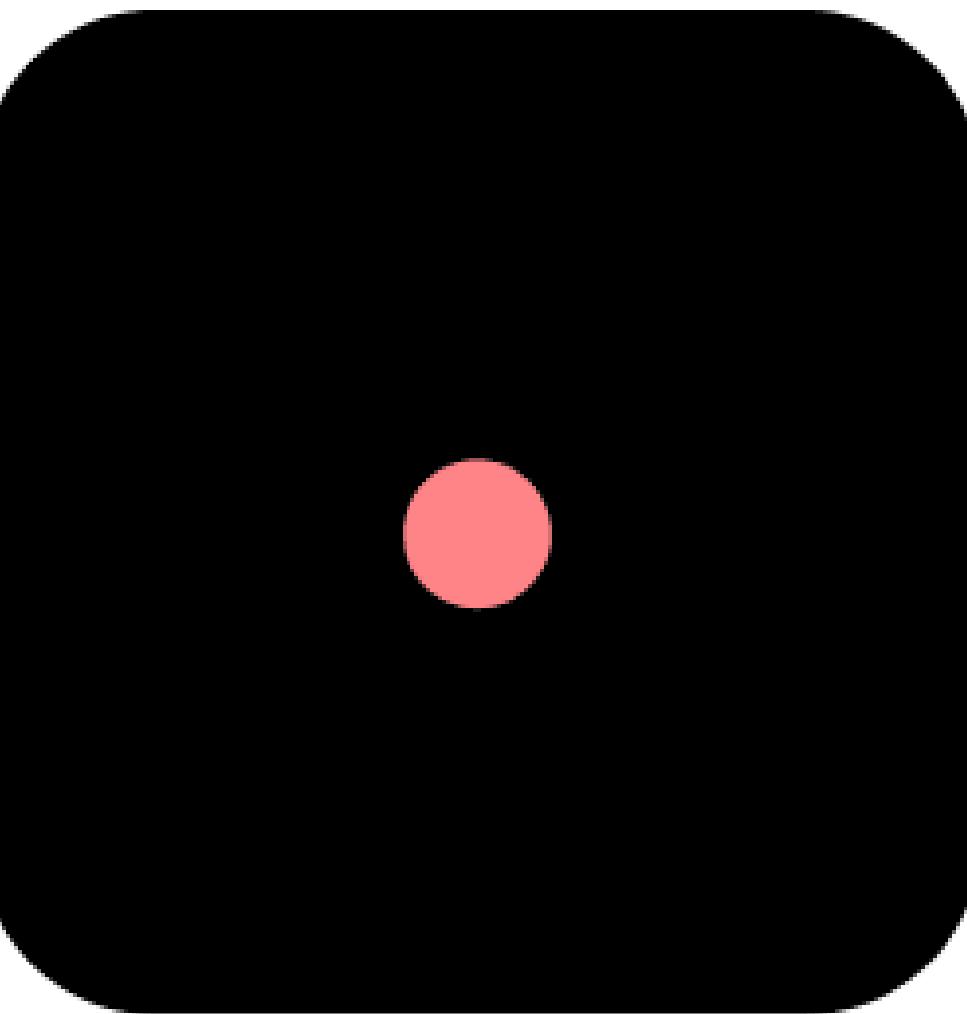
# Principios básicos

## Bitmap images vs Vectors

Vectors



Images



# Principios básicos

Artículo completo

# Preparación previa

# Preparación previa

- ¿Cuál es tu público objetivo?
- ¿Móvil o escritorio o ambos?
- Conoce las resoluciones más utilizadas de tu público potencial, para eso tenemos Google Analytics
- Elige una estrategia acorde a los datos anteriores

# Preparación previa

Comparador de resoluciones

# Preparación previa

## Cuidado con Google Analytics

Sólo mira analíticas de sitios que estén  
preparados para todas las resoluciones

# Preparación previa

También es aconsejable echar un vistazo a información externa como las que nos proporcionan estadísticas globales anónimas de Global StatCounter, para hacernos una idea de los atributos más comunes.

# Preparación previa

Una vez tengamos decidida una estrategia, el departamento de diseño la aplicará desde el inicio del proyecto

# Viewport

# Viewport

En muchos casos puede que oigas hablar del viewport del navegador. Esa palabra hace referencia a la región visible del navegador, o sea, la parte de la página que está visualizándose actualmente en el navegador.

# Viewport

Los usuarios podemos redimensionar la ventana del navegador para reducir el tamaño del viewport y simular que se trata de una pantalla y dispositivo más pequeño

# Viewport

Si queremos editar ciertos comportamientos del viewport del navegador, podemos editar el documento HTML para especificar el siguiente campo meta, antes de la parte del </head>:

```
<meta name="viewport" content="initial-scale=1, width=device-width">
```

# Viewport

Parámetro	Valor	Significado
width	device-width	Indica un ancho para el viewport.
height	device-height	Indica un alto para el viewport.
initial-scale	1	Escala inicial con la que se visualiza la página web.
minimum-scale	0.1	Escala mínima a la que se puede reducir al hacer zoom.
maximum-scale	10	Escala máxima a la que se puede aumentar al hacer zoom.
user-scalable	no o fixed   yes o zoom	Posibilidad de hacer zoom en la página web.

# Viewport

```
<meta name="viewport" content="initial-scale=1, width=device-width">
```

# Viewport

En dispositivos móviles, el viewport cambia de tamaño y se mueve cuando el usuario hace zoom o cambia la orientación de su dispositivo

# Viewport

Con esto conseguiremos preparar nuestra web para dispositivos móviles y prepararnos para la introducción de reglas media query en el documento CSS.

**Es importante no olvidar este paso.**

## Conclusión

La gestión del viewport es crucial para el diseño responsive, ya que permite a los diseñadores y desarrolladores controlar cómo se adapta y se muestra el contenido en diferentes tamaños de pantalla, asegurando una experiencia de usuario coherente y accesible.

# Instrucción @media

Una vez nos adentramos en el mundo del Responsive Design, nos damos cuenta en que hay situaciones en las que determinados aspectos o componentes visuales deben aparecer con ciertas diferencias dependiendo del dispositivo donde se están visualizando, ya que no todos los dispositivos tienen los mismos tamaños o características

Por ejemplo, una zona donde se encuentra el buscador de la página puede estar colocada en un sitio concreto en la versión de escritorio, pero en móvil quizás nos interese que ocupe otra zona (o que tenga otro tamaño o forma) para aprovechar mejor el espacio de la versión del dispositivo móvil.

¿Qué son las media queries?

Existe un concepto denominado media queries, mediante el cuál podemos hacer excepciones para que unos determinados estilos de diseño sólo se apliquen si se cumplen una serie de condiciones, generalmente relacionadas con el dispositivo mediante el cuál se está viendo la página.

Regla	Descripción
@media (<condición>)	Si se cumple la condición, se aplican los estilos de su interior.
@media not (<condición>)	Si no se cumple la condición, se aplican los estilos de su interior.

Básicamente, se trata de establecer una condición que se aplicará a la página actual, y en el caso de cumplirse, se aplicarán los estilos indicados en su interior. De no cumplirse, no se aplicarán. Si utilizamos la keyword `not` antes de la condición, se invierte la condición

```
@media (*condición*) {  
    .container {  
        background: green;  
    }  
}  
  
@media not (*condición*) {  
    .container {  
        background: red;  
    }  
}
```

En el ejemplo anterior, si se cumple la condición establecida, se aplicará un color verde. Sin embargo, si no se cumple, se aplicará un color rojo. Observa que es similar al funcionamiento de un if / else en programación

```
@media (*condición*) {  
    .container {  
        background: green;  
    }  
}  
  
@media not (*condición*) {  
    .container {  
        background: red;  
    }  
}
```

En el ejemplo anterior, si se cumple la condición establecida, se aplicará un color verde. Sin embargo, si no se cumple, se aplicará un color rojo. Observa que es similar al funcionamiento de un if / else en programación

# Condiciones múltiples

```
@media (*condición*) and (*condición*) {  
    .container {  
        background: orangered;  
    }  
}
```

En el ejemplo anterior, si se cumple la condición establecida, se aplicará un color verde. Sin embargo, si no se cumple, se aplicará un color rojo. Observa que es similar al funcionamiento de un if / else en programación

# Tipos de “medios”

Tipo de medio	Significado
all	Todos los dispositivos o medios. El que se utiliza <b>por defecto</b> .
screen	Monitores o pantallas de ordenador. Es el más común.
print	Documentos de <u>medios impresos</u> o pantallas de previsualización de impresión.
speech	Lectores de texto para invidentes (Antes aural, el cuál ya está obsoleto).

## Tipo de medio y condición

```
@media screen and (*condición*) {  
    /* ... */  
}  
  
@media print {  
    /* ... */  
}
```

## ¿Qué podemos meter como condición?

En los ejemplos anteriores hemos indicado \*condiciones\* en el interior de los paréntesis, pero no hemos visto como definir ninguna. Vamos a echar un vistazo a que tipo de características podemos definir en las condiciones de los media queries.

# ¿Qué podemos meter como condición?

Tipo de característica	Valores	¿Cuándo se aplica?
width	SIZE	Si el dispositivo tiene el <b>tamaño de ancho exactamente</b> .
min-width	SIZE	Si el dispositivo tiene un <b>tamaño de ancho mayor</b> al indicado.
max-width	SIZE	Si el dispositivo tiene un <b>tamaño de ancho menor</b> al indicado.
height	SIZE	Si el dispositivo tiene el <b>tamaño de alto exactamente</b> .
min-height	SIZE	Si el dispositivo tiene un <b>tamaño de alto mayor</b> al indicado.
max-height	SIZE	Si el dispositivo tiene un <b>tamaño de alto menor</b> al indicado.
aspect-ratio	<i>aspect-ratio</i>	Si el dispositivo encaja con la <b>proporción de aspecto</b> indicada.
orientation	landscape   portrait	Si el dispositivo está en colocado en <b>modo vertical o apaisado</b> .

## ¿Qué podemos meter como condición?

```
.container {  
    background: grey;  
    height: 100px;  
}  
  
@media (max-width: 800px) and (orientation: landscape) {  
    .container {  
        background: red;  
    }  
}
```



¿Lo probamos?

- Azul para resoluciones menores a 400 píxeles de ancho (móviles).
- Rojo para resoluciones entre 400 píxeles y 800 píxeles de ancho (tablets).
- Verde para resoluciones mayores a 800 píxeles (desktop).

## Media query range

Aunque la forma de definir condiciones anterior es útil, a menudo resulta confusa y difícil de escribir. Es posible utilizar una modalidad denominada rangos de condiciones, que son algo más versátiles que las anteriores y mucho menos tediosas.

## Media query range

La diferencia radica en que se puede utilizar operadores matemáticos que harán que las condiciones sean mucho más legibles y fáciles de interpretar. Veamos el mismo ejemplo anterior con sintaxis de rangos:

# @media

```
@media (width ≤ 400px) {  
  .menu {  
    background: blue;  
  }  
}  
  
@media (400px ≤ width ≤ 800px) {  
  .menu {  
    background: red;  
  }  
}  
  
@media (width ≥ 800px) {  
  .menu {  
    background: green;  
  }  
}
```

## Media query range

## Media desde el html

Por último, hay que tener en cuenta que los media queries también es posible indicarlos desde HTML, utilizando la etiqueta `<link>` y el atributo `media` para establecer la condición

## Media desde el html

```
<link rel="stylesheet"
      href="mobile.css"
      media="(max-width: 640px)">

<link rel="stylesheet"
      href="tablet.css"
      media="(min-width: 640px) and (max-width: 1280px)">

<link rel="stylesheet"
      href="desktop.css"
      media="(min-width: 1280px)">
```

# Breakpoints

# Breakpoints

Los breakpoints en el diseño web responsive  
son las **medidas específicas** donde el  
contenido y el diseño de un sitio web  
cambiarán para adaptarse a diferentes  
tamaños de pantalla

# Breakpoints

Estos breakpoints están comúnmente basados en las anchuras de los dispositivos y se definen utilizando la regla @media en CSS.

# Breakpoints

La teoría

# Breakpoints

## Teléfonos Móviles (pequeños)

320px: Ancho de pantalla para teléfonos móviles más antiguos.

360px - 400px: Ancho de pantalla para teléfonos móviles modernos.

# Breakpoints

## Teléfonos Móviles (grandes)

414px - 480px: Pantallas más grandes de dispositivos móviles y phablets.

# Breakpoints

## Tabletas (verticales/portrait)

600px - 768px: Ancho de pantalla para tabletas pequeñas y dispositivos intermedios.

# Breakpoints

**Tabletas (horizontales/landscape) /**

**Pantallas pequeñas de laptops**

768px - 1024px: Ancho de pantalla para tabletas en orientación horizontal y pantallas pequeñas de laptops.

# Breakpoints

## Laptops / Desktops

1024px - 1280px: Ancho de pantalla para laptops y monitores pequeños.

1280px - 1366px: Un breakpoint común para laptops de tamaño estándar.

# Breakpoints

## Desktops (grandes)

1440px - 1600px: Pantallas de desktops de alta resolución.

# Breakpoints

**Desktops (extra grandes) / Monitores de TV:**

1920px y más: Para pantallas de alta

resolución, incluyendo monitores 4K.

# Breakpoints

**Optimiza tus esfuerzos**

# Breakpoints

Es importante tener en cuenta que estos valores pueden variar según el público objetivo y los dispositivos más comunes entre los usuarios del sitio web. Además, con la variedad de dispositivos en constante cambio, es recomendable usar un enfoque de diseño web "first mobile" o "mobile-first", donde se comienza con el diseño para móviles y luego se escalan los estilos para pantallas más grandes.

# **Mobile First vs Desktop First**

# Mobile First vs Desktop First

¿Por qué es mejor usar Mobile First en lugar  
de Desktop First?

# Mobile First vs Desktop First

Les obliga a los diseñadores a pensar

# Mobile First vs Desktop First

Prioridad al usuario móvil

# Mobile First vs Desktop First

Esencial y útil vs Visual

# Mobile First vs Desktop First

Mejora la velocidad de carga

=

mejor posicionamiento SEO

# Mobile First vs Desktop First

100% la UX estará mejor pensada y adaptada  
de pequeño a grande que si lo hacemos al  
contrario

# Mobile First vs Desktop First

Desarrollo progresivo, siempre mejor,  
tanto para pensar como para  
maquetar/programar

# Mobile First vs Desktop First

Compatibilidad de serie con el “Tap”

# Mobile First vs Desktop First

Google y el  
Progressive Web Apps (PWA)  
mandan...

Hagámosle caso

**min-max-width-height  
min-max-content**

**min - max**

**W3C**

# <Despedida>

Email

**bienvenidosaez@gmail.com**

Instagram

**@bienvenidosaez**

Youtube

**youtube.com/bienvenidosaez**

**CONQUERBLOCKS**