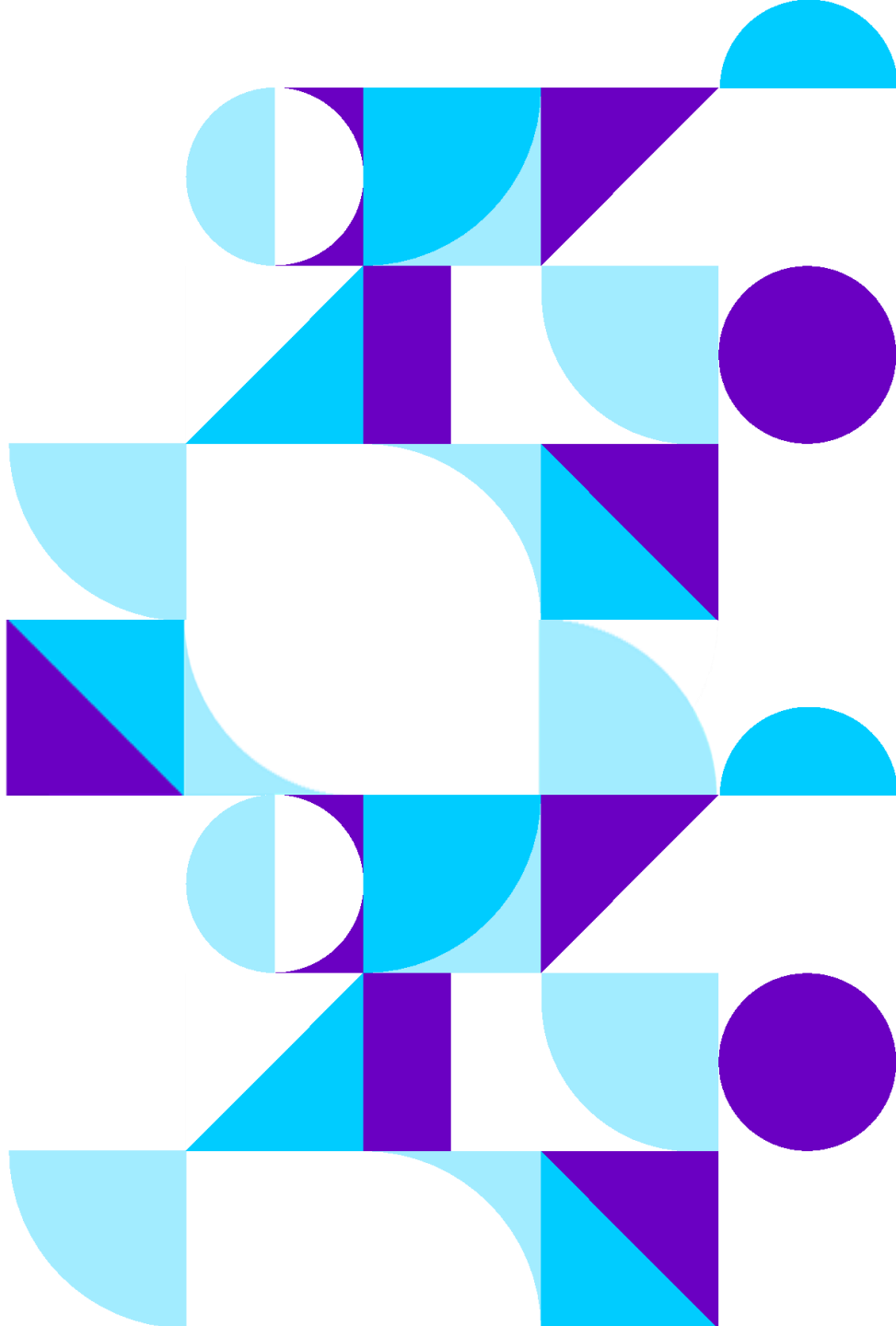
edix™

DAW_Diseño de Interfaces Web

edix™

UF3 – Programación



Contenido

- Imágen, vídeo, audio
- Transiciones, animaciones, sombras
- Media queries
- FlexBox y Grid

Imágenes. Tipos

- **Imagen vectorial**: trabajan independientemente de la resolución, por lo que mantienen la nitidez y de nición si las ampliamos o reducimos: nunca perderán la calidad. PSD, PDF.

- **Mapa de bits**: matriz cartesiana (bidimensional) que determina las coordenadas (horizontales y verticales) de la posición de cada píxel en la imagen. Requieren siempre más memoria que los vectores. BMP, GIF, JPG, PNG.

La optimización de la imagen es fundamental, porque va a permitirnos un mejor rendimiento

Los formatos más utilizados en la web son:

- png: si la imagen tiene texto o si necesitamos un fondo transparente.
- gif: para imágenes muy pequeñas.
- jpg: visualización de fotos, ilustraciones, imágenes, etc

Usar webfonts

```
@font-face {
  font-family: 'Tagesschrift';
  src: url('tagesschrift.eot');
       url('tagesschrift.woff') format('woff'),
       url('tagesschrift.ttf') format('truetype'),
       url('tagesschrift.svg#font') format('svg');
}
```

```
p { font-family: "Tagesschrift", Georgia, Serif; }
```

Audio

```
<audio controls="">
      <source src="foo.opus" type="audio/ogg; codecs=opus"/>
      <source src="foo.ogg" type="audio/ogg; codecs=vorbis"/>
      <source src="foo.mp3" type="audio/mpeg"/>
</audio>
```

- ***src***: URL del audio a reproducir, obligatoria si actúa como etiqueta contenedora.
- ***preload***: indica cómo realizar la precarga del audio (*auto*, *metadata* y *none*).
- ***mediagroup***: establece un nombre para un grupo de contenidos multimedia.
- ***autoplay***: reproduce el audio automáticamente.
- ***loop***: vuelve a iniciar el audio cuando finaliza su reproducción, es decir, crea un bucle.
- ***muted***: establece un silenciado al sonido.
- ***controls***: muestra los controles de reproducción, aunque por defecto no se muestran.

Vídeo

```
<video>
  <source src="pelicula.mp4" type='video/mp4; codecs="avc1.42E01E, mp4a.40
  <source src="pelicula.webm" type='video/webm; codecs="vp8, vorbis"' />
</video>
```

- ***src***: URL del vídeo a reproducir, obligatoria si actúa como etiqueta contenedora.
- ***preload***: indica cómo realizar la precarga del vídeo (*auto*, *metadata* y *none*).
- ***poster***: muestra una imagen a modo de presentación fija del vídeo.
- ***mediagroup***: establece un nombre para un grupo de contenidos multimedia.
- ***autoplay***: reproduce el vídeo automáticamente.
- ***loop***: vuelve a iniciar el vídeo cuando finaliza su reproducción, es decir, crea un bucle.
- ***muted***: establece un silenciado al sonido.
- ***controls***: muestra los controles de reproducción, aunque por defecto no se muestran.
- ***width***: indica el ancho del vídeo.
- ***height***: indica el alto del vídeo.

Transiciones

```
# capa { transition: <property> <duration> <timing-function> <delay>}
```

transition-property

Especificaremos la propiedad a la que afectará la transición .
Valores:

- *all*: hace que se aplique a todos los elementos con los que se encuentre.
- *none*: hace que no se aplique ninguna transición.
- *propiedad_determinada*: especifica la propiedad concreta (*width*, *height*, color...).

transition-duration

Es la duración de la transición, desde su inicio hasta su finalización.

Transiciones

```
# capa { transition: <property> <duration> <timing-function> <delay>}
```

transition-timing-function

Permite indicar el tipo de transición a conseguir.
Valores:

- *ease*: comienzo lento, luego rápido y termina lento.
- *linear*: mantiene la misma velocidad de principio a fin.
- *ease-in*: comienza lento, y después mantiene la velocidad.
- *ease-out*: mantiene la velocidad con un final lento.
- *ease-in-out*: comienzo y fin lentos, muy similar a *ease* solo que este último empieza más rápido de lo que termina.
- *cubic-bezier*(A, B, C, D): es una función personalizada donde se dan valores concretos depediendo de la velocidad que tenga la transición, donde A y B son los los primeros puntos que orientan la curva bezier (X e Y), y C y D son los segundos puntos que orientan la misma curva (X e Y).

Animaciones

```
# capa { animation: <name> <duration> <timing-function> <delay> <iteration
```

animation-name

Especifica el nombre de la regla ***@keyframes*** que describe los fotogramas de la animación.
Valores:

- *None*.
- Nombre_animación.

```
@keyframe nombre {
            selector_keyframe{
                        propiedad: valor;
                        propiedad: valor;
                        propiedad: valor;
            }
}
```

animation-duration

Cantidad de tiempo que la animación consume en completar su ciclo (*duración*).
Valores:

- 0.
- Tiempo_determinado.

Animaciones

```
# capa { animation: <name> <duration> <timing-function> <delay> <iteration
```

animation-timing-function

Ritmo de la animación, es decir, como se muestran sus fotogramas, estableciendo curvas de aceleración.
Valores:

- *Ease*.
- *Linear*.
- *Ease-in*.
- *Ease-out*.
- *Ease-in-out*.
- *Cubic-bezier* (A, B, C, D).

Animaciones

```
# capa { animation: <name> <duration> <timing-function> <delay> <iteration
```

animation-delay

Tiempo de retardo entre la carga del elemento y el comienzo de la secuencia de la animación.
Valores:

- 0.
- Tiempo_determinado.

animation-iteration-count

El número de veces que se repite. Si se indica ***infinite***, se repetirá de forma indefinida.
Valores:

- 1.
- *Infinite*.
- Número _determinado_iteracciones.

Animaciones

```
# capa { animation: <name> <duration> <timing-function> <delay> <iteration
```

animation-direction

Indica si la animación debe retroceder hasta el fotograma de inicio al finalizar la secuencia, o bien comenzar desde el principio al llegar al final.
Valores:

- *Normal*: los fotogramas se reproducen desde el principio al final.
- *Reverse*: los fotogramas se reproducen desde el final al principio.
- *Alternate*: en iteraciones par, de forma normal. En impares, a la inversa.
- *Alternate-reverse*: en iteraciones impares, de forma normal. En pares, normal.

Animaciones

```
# capa { animation: <name> <duration> <timing-function> <delay> <iteration
```

animation-fill-mode	Specifies a style for the element when the animation is not playing (before it starts, after it ends, or both)

Value	Description
none	Default value. Animation will not apply any styles to the element before or after it is executing
forwards	The element will retain the style values that is set by the last keyframe (depends on animation-direction and animation-iteration-count)
backwards	The element will get the style values that is set by the first keyframe (depends on animation-direction), and retain this during the animation-delay period

Sombras CSS

```
# capa {
    box-shadow: 2px 2px 10px #666;              /* Sombra normal */
    box-shadow: 5px -5px 0 2px #444;            /* Sombra superior sin des
    box-shadow: 5px 5px 25px #222 inset;        /* Sombra interior */
}
```

```
p {
    text-shadow: 2px 2px 0 #AAA,        /* Sombra 1 */
    6px 6px 2px #777,                   /* Sombra 2 */
    12px 12px 8px #444;                 /* Sombra 3 */
}
```

Viewport

Con esta propiedad hacemos referencia a la parte visible del navegador actual.

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

- ***width***: indica un ancho para el *viewport*, es decir, el *device-width*.
- ***height***: indica un alto para el *viewport*, es decir, el *device-height*.
- ***initial-scale***: escala inicial con la que se visualiza la web, con 1 como valor por defecto.
- ***minimum-scale***: escala mínima a la que se puede reducir al hacer zoom, con 0'1 como valor por defecto.
- ***maximum-scale***: escala máxima a la que se puede aumentar al hacer zoom, con 10 como valor por defecto.
- ***user-scalable***: posibilita hacer o no zoom sobre la web por parte del usuario en la pantalla de su dispositivo.

Media Queries: @media

Usando media queries se puede saber qué dispositivo se usa o conocer la resolución.

```
/* Para 960px */
@media only screen and (max-width: 980px) and (min-width: 821px) {  ...  }

/* Para 800px */
@media only screen and (max-width: 820px) and (min-width: 621px)  {  ...  }

/* Para 600px */
@media only screen and (max-width: 620px) and (min-width: 501px)  {  ...  }

/* Para 480px */
@media only screen and (max-width: 500px) and (min-width: 341px)  {  ...  }

/* Para 320px */
@media only screen and (max-width: 340px) and (min-width: 5px)   {  ...  }
```

Media Queries: @media

Las propiedades que más nos interesan son las siguientes:

a) width y height:

*Ancho y alto del navegador (podemos añadir el prefijo **min**– o **max**-)*

b) device-width y device-height:

*Ancho y alto del dispositivo, móviles y tablets (podemos añadir el prefijo **min**– o **max**-)*

c) orientation:

*Orientación del móvil o tablet (para panorámico utilizaremos **orientation:portrait**, para vertical **orientation:landscape**)*

CSS Flexbox

a guide from CSS-TRICKS

container

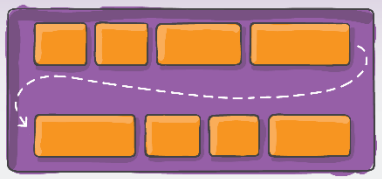
items

```
.container {
  display: flex; /* or inline-flex */
}
```

flex-direction

```
.container {
  flex-direction: row | row-reverse |
  column | column-reverse;
}
```

flex-wrap



```
.container {
  flex-wrap: nowrap | wrap | wrap-reverse;
}
```

align-content

flex-start

flex-end

center

stretch

space-between

space-around

```
.container {
  align-content: flex-start | flex-end |
  center | space-between | space-around |
  space-evenly | stretch;
}
```

order

1 1 1 2 3

-1 1 2 5

2

2

99

```
.item {
  order: 5; /* default is 0 */
}
```

justify-content

flex-start

flex-end

center

space-between

space-around

space-evenly

```
.container {
  justify-content: flex-start | flex-end |
  center | space-between | space-around |
  space-evenly;
}
```

align-items

flex-start

flex-end

center

stretch

baseline

text text text text text text

```
.container {
  align-items: stretch | flex-start |
  flex-end | center | baseline;
}
```

flex-shrink, flex-grow, flex-basis

1 1 1

1 2 1

50px 1 1

```
.item {
  flex-shrink: 1; /* default is 1 */
  flex-grow: 2; /* default is 0 */
  flex-basis: 50px; /* default auto */
}
```

align-self

stretch

flex-start

flex-end

```
.item {
  align-self: auto | flex-start |
  flex-end | center | baseline | stretch;
}
```

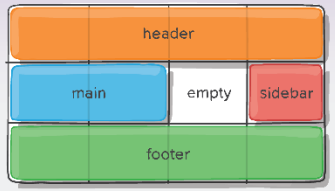
CSS Grid

a guide from CSS-TRICKS

grid-template-columns grid-template-rows

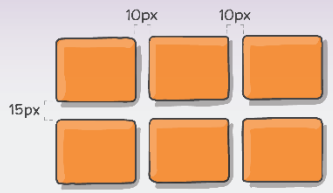

```
.container {
  grid-template-columns: <value> | <name>;
  grid-template-rows: <value> | <name>;
}
```

grid-template-areas




```
.container {
  grid-template-areas: "<name> | . | none";
}
```

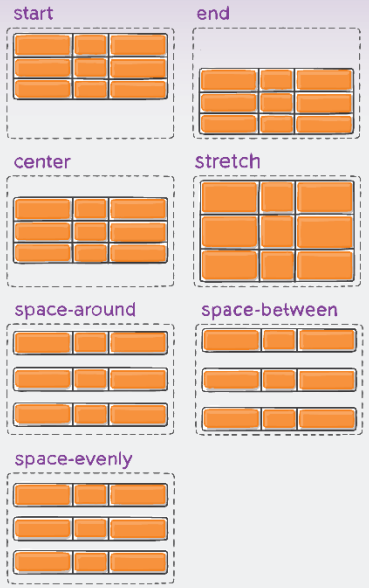
column-gap, row-gap, gap




```
.container {
  column-gap: <value>;
  row-gap: <value>;
  gap: <row-gap> <column-gap>;
}
```

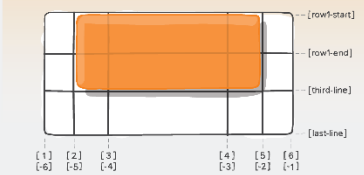
```
.container {
  display: grid; /* or inline-grid */
}
```

align-content



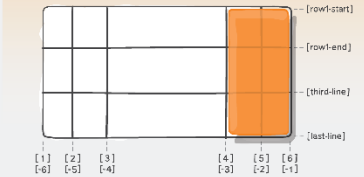

```
.container {
  align-content: start | end | center |
  stretch | space-around | space-between |
  space-evenly;
}
```

grid-column, grid-row



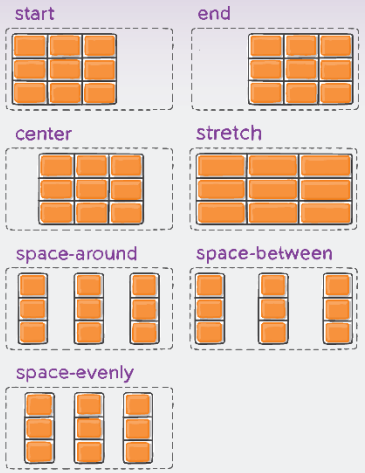

```
.item {
  grid-column: <start-line> / <end-line> |
  <start-line> / span <value>;
  grid-row: <start-line> / <end-line> |
  <start-line> / span <value>;
}
```

grid-area



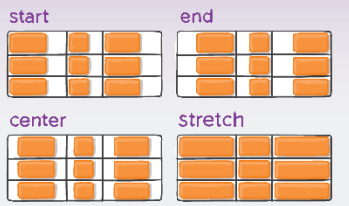

```
.item {
  grid-area: <row-start> / <column-start>
  / <row-end> / <column-end> | <name>;
}
```

justify-content



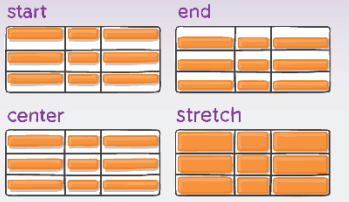

```
.container {
  justify-content: start | end | center |
  stretch | space-around | space-between |
  space-evenly;
}
```

justify-items



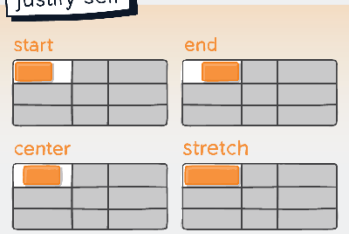

```
.container {
  justify-items: start | end |
  center | stretch;
}
```

align-items



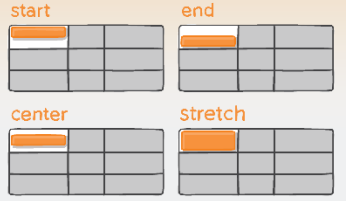

```
.container {
  align-items: start | end |
  center | stretch;
}
```

justify-self




```
.item {
  justify-self: start | end | center |
  stretch;
}
```

align-self




```
.item {
  align-self: start | end | center |
  stretch;
}
```

Instituto Tecnológico Edix_