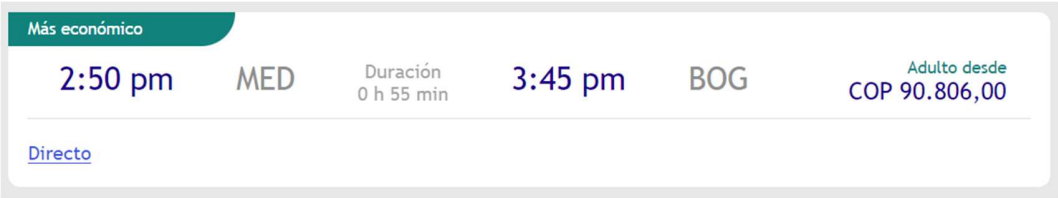
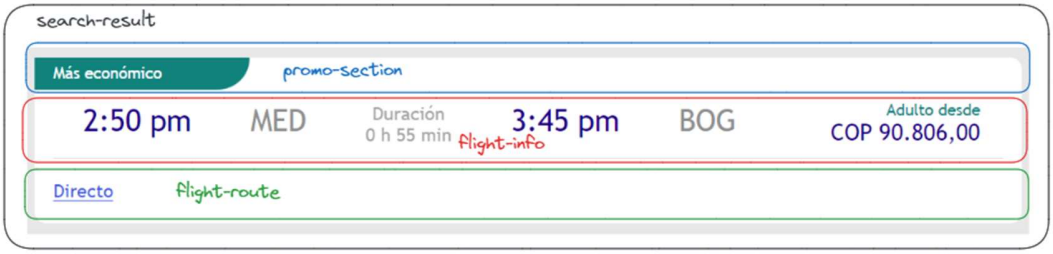
03 Caja de resultados – Estilos

Vamos a agregar el siguiente código a nuestro html, el cual usaremos para presentar cada elemento de los resultados de búsqueda. El resultado final que queremos obtener es algo como lo siguiente:

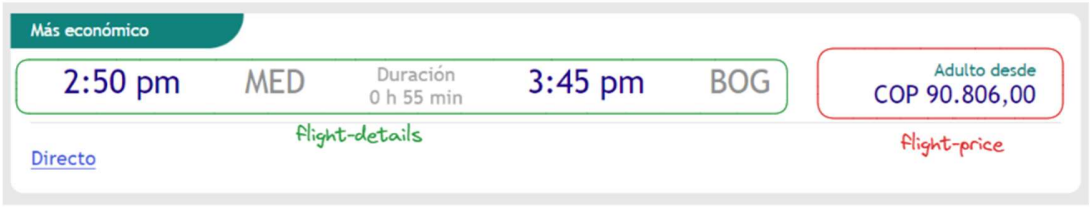


En este caso y para entender mejor el ejercicio, lo primero que haremos será colocar diferentes etiquetas ***div*** para estructurar cada sección de los resultados. El contenedor principal lo marcaremos con la clase ***search-result***, esta sección estará dividido en tres partes, diferenciadas también por clases: ***promo-section***, ***flight-info*** y ***flight-route***.



La primera sección, ***promo-section***, la usaremos para agregar un efecto especial a los vuelos que tengan un precio especial, esto lo haremos con la ayuda de una etiqueta ***span*** y una clase ***promo-label*** que veremos en detalle más adelante.

La siguiente sección, ***flight-info*** a su vez se divide en dos más, ***flight-details*** y ***flight-price***, la primera mostrará la hora y lugar de salida, al igual que hora y lugar de llegada, además de la duración del vuelo; la segunda mostrará el precio del vuelo.



La última sección, ***flight-route***, mostrará la cantidad de paradas que pueda tener el vuelo o la palabra ***Directo***, en caso de no tener paradas intermedias. Por ahora el html se vería así:

```
<section>
  <section class="search-result">
    <div class="promo-section">
      <span class="promo-label">Más económico</span>
    </div>
    <div class="flight-info">
      <div class="flight-details"></div>
      <div class="flight-price"></div>
    </div>
    <div class="flight-route"></div>
  </section>
</section>
```

Vamos a colocar algunos valores en cada sección, además definiremos unas cajas más. Tendremos también unas secciones adicionales:

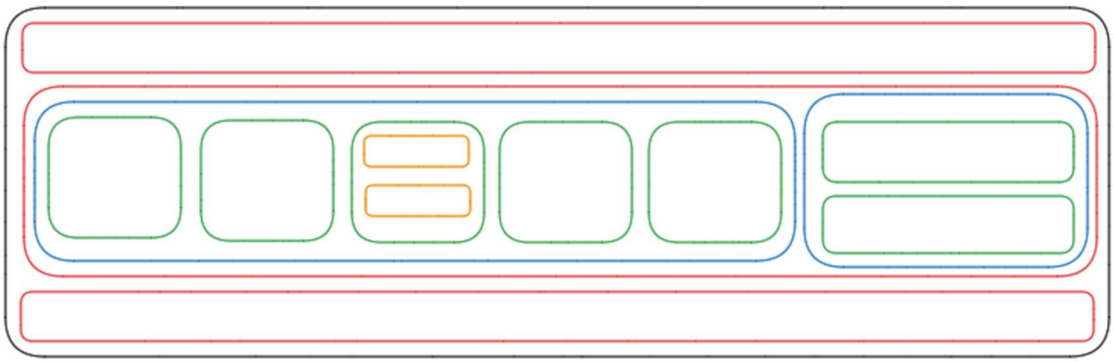
Dentro del ***flight-***info agregaremos el ***flight-time***, que nos servirá para dar estilo a las horas de salida y llegada; el ***flight-location*** lo usaremos para la ciudad de origen y de destino; finalmente, el ***duration-info*** contendrá la información de la duración del viaje.

Mientras que en el ***flight-price***, tendremos un pequeño componente con clase ***Price*** para dar formato al valor del tiquete.

En este punto es importante ver la relación entre los componentes, quien contiene a quien ya que con esto formaremos algo como una cuadrícula o grilla (***grid***) donde iremos posicionando los diferentes textos.

```
<section>
  <section class="search-result">
    <div class="promo-section">
      <span class="promo-label">Más económico</span>
    </div>
    <div class="flight-info">
      <div class="flight-details">
        <div class="flight-time">2:50 pm</div>
        <div class="flight-location">med</div>
        <div class="duration-info">
          <div>Duración</div>
          <div>0 h 55 min</div>
        </div>
        <div class="flight-time">3:45 pm</div>
        <div class="flight-location">bog</div>
      </div>
      <div class="flight-price">
        <div>Adulto desde</div>
        <div class="price">COP 90.806,00</div>
      </div>
    </div>
    <div class="flight-route">
      <a href="#">Directo</a>
    </div>
  </section>
</section>
```

Esta cuadrícula se vería más o menos de la siguiente forma:



Lo importante de entender este esquema es poder observar que tendremos un contenedor principal, dentro del cual colocaremos tres cajas que se distribuirán de manera vertical. La caja de en medio contendrá a su vez dos cajas, dispuestas de manera horizontal; la primera de estas cajas se dividirá en 5 cajas más, distribuidas también de manera horizontal, conteniendo la caja del medio dos cajas horizontalmente; y la segunda contendrá otras dos cajas dispuestas de manera vertical. Vemos entonces que casi todos los elementos están contenidos dentro de otro, excepto el contenedor principal, y, a su vez, contienen algún otro componente. Para lograr esta distribución utilizaremos dentro de las clases que definamos, la propiedad ***display***, la cual ya hemos usado para modificar el comportamiento en línea o en bloque de los elementos, pero en esta ocasión le asignaremos el valor de ***flex***, lo cual dispondrá cada elemento uno a continuación del otro. Para determinar en qué sentido se dispondrán los componentes es necesario incluir la propiedad ***flex-direction***, la cual puede tener los valores ***row*** (disposición horizontal) o ***column*** (disposición vertical), el valor por defecto es ***row***, así que si no especificamos esta propiedad los elementos se distribuirán horizontalmente.

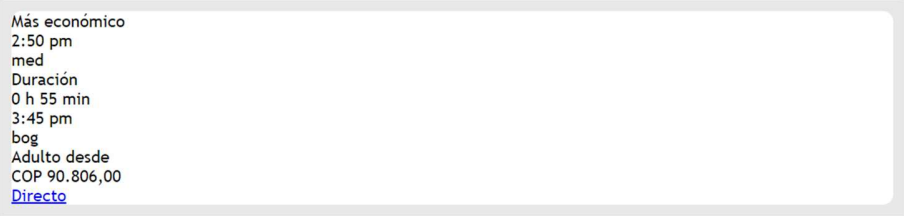
Como aún no hemos definido ninguna clase, deberíamos estar observando lo siguiente:

Más económico
2:50 pm
med
Duración
0 h 55 min
3:45 pm
bog
Adulto desde
COP 90.806,00
Directo

Empecemos entonces con los estilos. Definiremos primero el contenedor principal, al cual le asignamos la clase ***search-result***, con las siguientes propiedades, de las cuales conocemos la mayoría y nos enfocaremos en explicar las nuevas:

```css
.search-result{
  background: #fff;
  width: 80%;
  max-width: 1000px;
  margin: 10px auto;
  border-radius: 10px;
  display: flex;
  flex-direction: column;
}

.search-result:hover{
  box-shadow: rgba(16, 0, 79, 0.16) 0px 8px 16px 0px;
}
```

En esta clase estamos definiendo un color de fondo y un ancho, el cual tendrá un valor máximo impidiendo que el contenedor crezca indefinidamente con la pantalla. La propiedad ***flex***, como ya lo mencionamos, servirá para que todos los componentes dentro de este contenedor se ubiquen uno a continuación del otro. Como queremos que la distribución sea vertical, especificamos la propiedad ***flex-direction: column***. Definimos también para la clase ***search-result*** la pseudo clase ***hover***, con la propiedad ***box-shadow*** para dar efectos de sobra al contenedor, esto resaltará la caja cada vez que pasemos sobre ella. Los valores que asignamos a esta propiedad indican el color de la sombra y su transparencia, al igual que el tamaño de la misma[1]. Por ahora aún no vemos cambios respecto a la distribución, ya que todo estaba dispuesto de manera vertical.

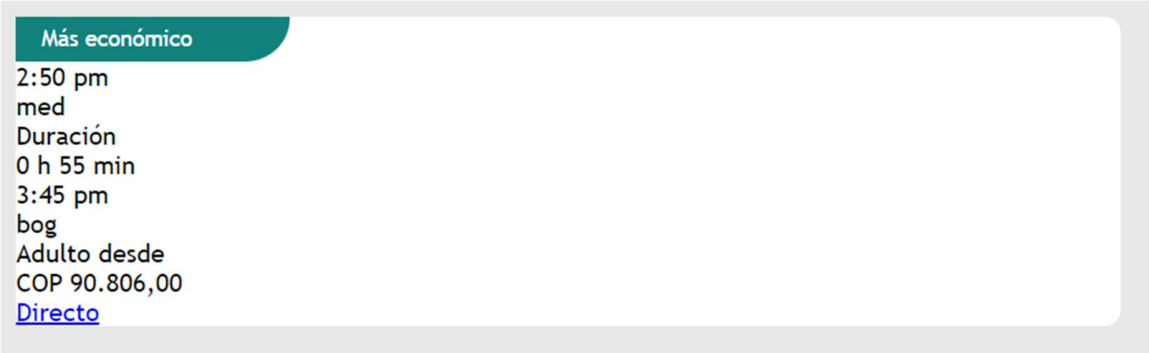


En la clase ***promo-section***, simplemente definiremos la altura que ocupará. Dentro de ella tendremos una etiqueta ***span*** con un texto que se visualizará o no, dependiendo de si el vuelo tiene un mejor precio respecto a otros. A esta etiqueta le asignamos la clase ***promo-label***. Como la etiqueta ***span*** es de tipo en línea, debemos modificar si comportamiento con la propiedad ***display: block***, si no hacemos esto, el atributo ***padding*** no generaría el espaciamiento que buscamos alrededor del texto. Así como anteriormente definimos un ***max-width***, acá tenemos un ***min-width***, que nos permite establecer un ancho mínimo. También podemos generar un efecto simple jugando con la propiedad ***border-radius***

```css
.promo-section{
  height: 20px;
}

.promo-label {
  display: block;
  border-radius: 0px 0px 60px;
  width: 18%;
  min-width: 140px;
  padding: 6px 16px;
  font-size: 14px;
  color: #fff;
  background-color: #11837c
}
```

[1] Para más detalle de los valores de la propiedad ***box-shadow***, puede consultar: https://developer.mozilla.org/es/docs/Web/CSS/box-shadow

Nuestro componente se vería de la siguiente manera:

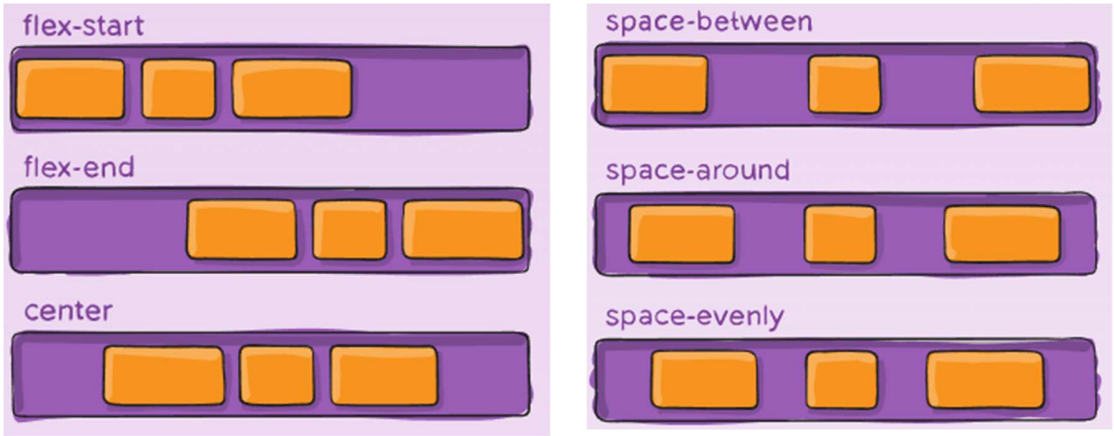


Ahora vamos a crear la clase ***flight-info***, según el esquema de cuadrícula visto anteriormente, es el componente del medio dentro de nuestro contenedor principal, además contendrá dos elementos que se van a disponer uno a continuación del otro:

```
.flight-info{
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin: 10px 16px;
  cursor: pointer;
}
```

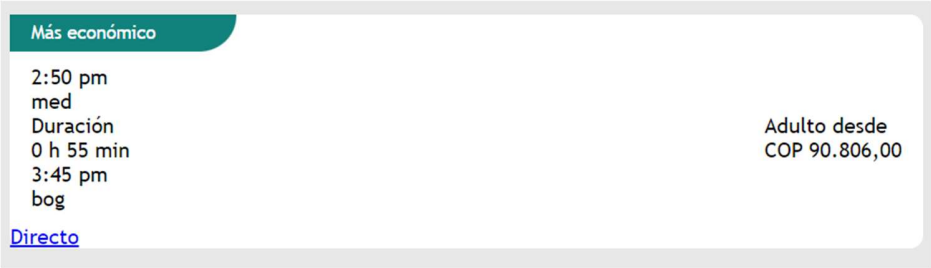
De nuevo utilizamos el ***display: flex*** para indicar que los elementos hijos de este contenedor se ubicarán uno a continuación del otro, pero como en este caso queremos que la distribución sea horizontal, no es necesario indicar la propiedad ***flex-direction***. La propiedad ***align-items*** nos permite definir la alineación en sentido vertical de los componentes.

La propiedad ***justify-content*** es muy útil ya que nos permite definir la separación de los componentes respecto al eje horizontal. Veamos un ejemplo de cómo se comporta según sus posibles valores[2]:



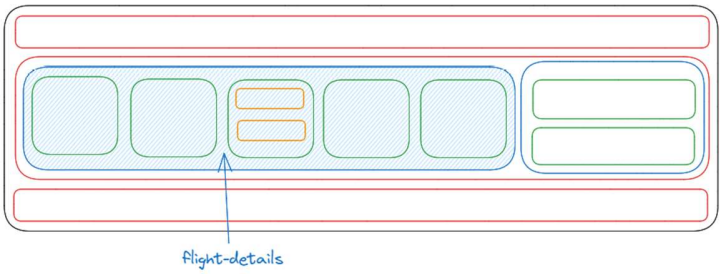
En nuestro caso la propiedad ***justify-content: space-between***, los componentes se ubican en los extremos y el espacio restante se ubica en el medio:

[2] La imagen fue tomada de https://css-tricks.com/snippets/css/a-guide-to-flexbox/, allí también se puede encontrar información más detallada de esta propiedad.

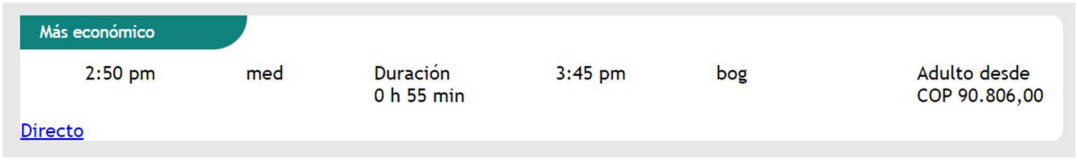


Ahora nos enfocaremos en el ***flight-details***

```css
.flight-details{
  display:flex;
  justify-content:space-around;
  width: 75%;
}
```



Este contenedor ocupará el 75% del ancho disponible, los elementos que contenga se distribuirán uno a continuación del otro con la propiedad ***flex***, nuevamente no es necesario indicar la orientación ya que por defecto será horizontal. Con ***justify-content: space-around***, los componentes dentro del contenedor se distribuirán uniformemente:



Modificamos las clases ***flight-time*** y ***flight-location*** para modificar la apariencia de estos textos. Para destacar en este punto, la propiedad ***transform-text:uppercase*** modificará cualquier texto que contenga y lo colocará en mayúscula sostenida. También podríamos usar un ***lowercase***, para presentar todo en minúsculas.

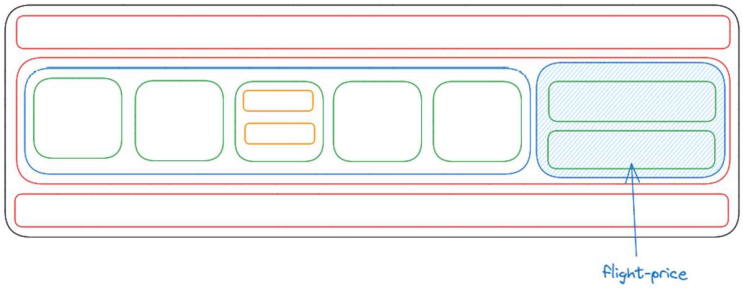
```css
.flight-time{
  font-size: 26px;
  color: #1b0088;
}

.flight-location{
  font-size: 26px;
  color:#8f8f8f;
  text-transform: uppercase;
}
```

La clase ***duration-info*** nos permite mostrar dos cajas, una sobre otra, esto lo logramos con la propiedad ***display: flex;*** acompañado de ***flex-direction: column***, para que los elementos se ubiquen uno sobre otro. La propiedad ***opacity*** nos da un texto un poco más claro.

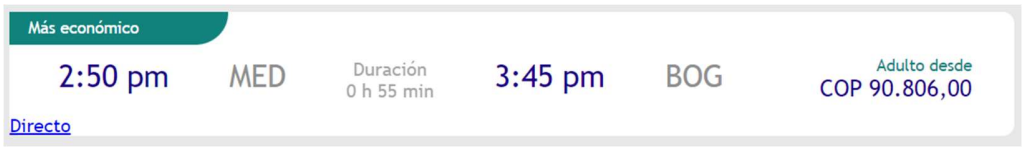
```css
.duration-info{
  display: flex;
  flex-direction: column;
  text-align: center;
  color: #8f8f8f;
  opacity: 0.8;
}
```

Ahora trabajaremos en las clases ***flight-price*** y ***price***,

```css
.flight-price{
  text-align: right;
  font-size: 14px;
  color: #11837c;
  margin-right: 20px;
}

.price{
  color: #1b0088;
  font-size: 20px;
}
```



Acá simplemente alinearemos los componentes a la derecha y modificaremos tamaño y color

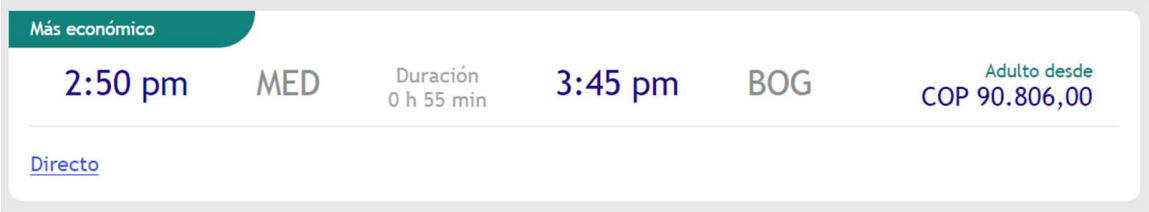


Por último, tenemos el contenedor con la clase ***flight-route***:

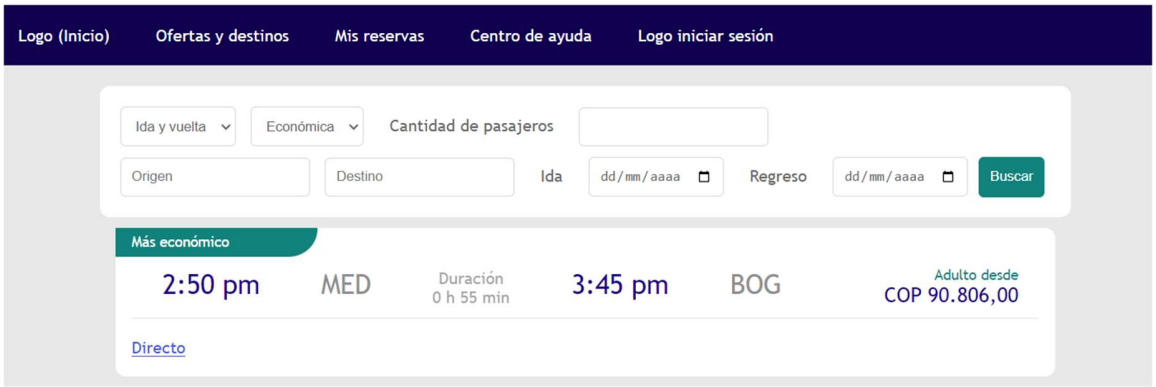
```css
.flight-route{
  padding-top: 16px;
  padding-bottom: 16px;
  margin: 0px 16px 0px;
  border-top: 1px solid rgb(225, 225, 225);
}

.flight-route a{
  color: #4658DF;
  line-height: 1.5;
  text-decoration: underline #4658DF 0.063rem;
  text-underline-offset: 2px;
}

.flight-route a:hover{
  text-decoration: none;
}
```

Acá daremos estilo, además, a los elementos ***a*** que se ubiquen dentro del contenedor con la clase ***flight-route***, también alteraremos el comportamiento al pasar con el mouse sobre el elemento ***a***.

De estas tres definiciones de estilo, mencionaremos algunos aspectos: el ***border-top*** nos mostrará una pequeña línea divisoria en el componente. El ***line-height*** permite definir una altura cuando tenemos texto, como si fuesen los espacios de separación de un párrafo. Con el ***text-decoration*** definiremos el color y grosor de la línea que subraya la etiqueta ***a***, mientras que con el ***text-underline-offset*** definiremos la separación entre el subrayado y el texto.



Hasta este punto deberíamos tener lo siguiente:



En este ejercicio tuvimos una amplia combinación de cajas (***div***) las cuales necesitábamos ubicar de cierta manera. La clave para lograr eso fue el uso de la propiedad ***display:flex***, más adelante veremos que esto también es clave para que nuestro diseño se ajuste al tamaño del navegador desde el que se visualice (responsive). Con el ***flex*** también estaremos habilitando otras propiedades que veremos más adelante.