Web publicada al FTP:

https://protected-content.ftp.uoc.edu/20221_20.450_html/aula1/rfogueras%40uoc.edu/practica_final/index.html
https://protected-content.ftp.uoc.edu/20221_20.450_html/aula1/rfogueras%40uoc.edu/practica_final/contact.html
https://protected-content.ftp.uoc.edu/20221_20.450_html/aula1/rfogueras%40uoc.edu/practica_final/color-theory-for-designers.html

Estructural HTML

L'estructura HTML de cada pàgina consta de tres parts principals, *<header>* *<main>* i *<footer>*.

- Dins del *<header>*, assignarem a *<nav>* la classe *.container*, que ens servirà per delimitar els límits d'amplada del contingut a través de la declaració CSS pertinent.
- Dins del *<main>* de cada pàgina hi ha diferents * * amb l'ID de cada apartat de la pàgina que ens servirà per indicar el color de fons de cada apartat, on dins hi haurà un element *<article>* o * * amb la classe *.container*, per delimitar l'amplada.
- Dins del *<footer>* seran dos * * que tindran la classe *.container* per delimitar l'amplada.

HEADER (apareix a cada pàgina):

El *<header>* consta d'un * * amb la classe *.skip-link* i un *<nav>* on ubicarem tots els elements del menú principal. Dins el * * ubicarem els enllaços per saltar al contingut i al peu de pàgina per millorar l'accessibilitat.

Dins del *<nav>* tenim dos elements principals:

1. Un enllaç *<a>* que conté un *<img>* amb el logo de l'empresa.
2. Una llista no ordenada <ul> amb els seus elements *<li>* i l'enllaç *<a>*.

A més, creem un *<input>* de tipus checkbox amb l'ID *#menu-btn* relacionat amb un *<label>* (amb *for="menu-btn"*) que ens serviran per obrir i tancar el menú en la vista mòbil (detalls a l'apartat Responsive al final del document).

MAIN (index.html):

El *<main>* de *index.html* té l'ID *#index* i ens serveix per recollir l'enllaç de *skip-to-content*. L'estructurem amb cinc contenidors * *:

1. El primer * * té l'ID *#tittle* i consta d'un *<article>* amb la classe *.container* on col·loquem el *<h1>* i un paràgraf *<p>*.

2. El segon * * té l'ID *#experience* que inclou un * * amb la classe *.container* i consta de tres contenidors:
    - Un *<section>* amb la classe *.summary* i conté un *<h2>* i dos paràgrafs *<p>*.
    - Un * * amb la classe *.gallery* i ens servirà per ubicar les imatges.
    - Un *<section>* amb la classe *.blog-article* que conté els enllaços i el resum de les entrades al blog. També hi ha un * * amb el botó per veure més articles.

3. El tercer * * té l'ID *#team* amb un *<article>* amb la classe *.container* que conté un *<h3>,* un *<h2>* i un * * que permet ubicar cada membre de l'equip. Aquest * * té la classe *.team-members*, i conté un * * per cada membre amb la classe *.member*, que al mateix temps consta de dos elements * * interns, un per la foto amb la classe *.photo* i un per la informació amb la classe *.info.* Aquest últim té un *<h4>* per el nom del membre de l'equip, i un paràgraf *<p>* per el seu càrrec.

4. El quart * * té l'ID *#testimonials* amb un *<article>* amb la classe *.container* que conté un *<h3>,* un *<h2>.*

5. El cinquè * * té l'ID *#testimoni* amb un * * intern amb la classe *.container*. Dins tenim un altre *div>* amb la classe *.box* que ens servirà per crear una caixa amb el text del testimoni en un paràgraf, i el nom de l'autor en un altre.

6. El sisè i últim * * té l'ID *#help* amb un *<article>* amb la classe *.container* que conté un * * amb la classe *.info* per col·locar un *<h2>* un paràgraf *<p>* i un enllaç *<a>* que porta a la pàgina *contact.html.* També hi ha un altre * * amb una imatge *<img>.*

MAIN (color-theory-for-designers.html):

El *<main>* de *color-theory-for-designers.html* té l'ID *#blog* i ens serveix per recollir l'enllaç de *skip-to-content*. En separem el contingut amb cinc * *:

1. El primer * * té l'ID #*tittle* amb un *<article>* amb la classe *.container*, i únicament ens serveix per col·locar un *<h1>* i un paràgraf *<p>*.

2. El segon * * té l'ID #*summary* amb un *<article>* amb la classe *.container*, i un altre * * intern amb la classe *.summary-content.* Aquest conté dos * .* Un amb la classe *.summary-author* que ens serveixen per col·locar la foto de l'autor amb el seu peu d'imatge amb un *<figure>, <img>* i *<figcaption>*, i un altre * * amb la classe *.summary-text* que ens permet col·locar el text en dos paràgrafs *<p>.*

3. El tercer * * té l'ID *#primarycolor*, amb un *<article>* amb la classe *.container*. *Dins* l'*<article>* tenim un *<h2>* i un * * amb la classe *.two-columns* per separa el contingut en dues columnes. La primera, en un * * amb la classe *.left-column* conté els diferents paràgrafs del text. El segon * * té la classe *.right-column* i conté dos *<figure>* per les fotos i els seus peu d'imatge.
   El quart * * té l'ID *#inbrief* amb un *<article>* amb la classe *.container*. Dins l'*<article>* hi ha un <h2>, un paràgraf *<p>,* i una taula amb *<table>*:

   La taula l'estructurem amb un *<thead>* que conté la capçalera amb un *<tr>* que conté 3 *<th>.* Cada un d'aquests *<th>* té l'atribut *scope="col"* per tal de representar la capçalera i relacionar-lo amb el contingut d'aquella columna.
   Posteriorment hi ha un *<tbody>* amb tres grups de *<tr>* per indicar les files. Cada *<tr>* conté un *<th>* principal amb l'atribut *scope="row"* per tal de representar la capçalera de la fila i relacionar-lo amb el contingut de cada una. A més hi ha 2 *<td>* amb el contingut que col·loquem amb paràgrafs *<p>*.

   Després de la taula, hi ha un paràgraf *<p>* amb un enllaç *<a>* apuntant a l'article del blog original.

4. El quart * * té l'ID *#ourblog* i dins un *<article>* amb la classe *.container.* Dins hi ha un *<h2>* amb el títol i tres * * per cada apartat amb les classes *.box-series*, *box-resources i .box-explore*. Cada apartat consta d'un *<h3>* i una llista no numerada *<ul>* amb les classes *series, resources* i *.explore* respectivament, amb diferents *<li>* que encepsulen els enllaços *<a>*.

MAIN (contact.html):

El <*main*> de *contact.html* té l'ID *#contact* i ens serveix per recollir l'enllaç de *skip-to-content*. Separem el seu contingut amb quatre <*div*>.

1. El primer <*div*> té l'ID #*tittle* amb un <*article*> amb la classe *.container*, i únicament ens serveix per col·locar un <*h1*> i un paràgraf <*p*>.

2. El segon <*div*> té l'ID *#map* amb un <*div*> amb la classe *.container*, i ens serveix per col·locar un <*iframe*> amb les coordenades del mapa.

3. El tercer <*div*> té l'ID #*contact-method* amb un <*article*> amb la classe *.container*, que ens servirà per col·locar les 3 caixes amb els diferents mètodes de contacte. La seva estructura consta d'un altre <*div*> amb la classe *.contact-icons* que el seu temps consta de les tres caixes per cada mètode amb un <*div*> individual:

   El primer <*div*> té la classe *phone*, l'altre *.email* i el darrer *.address.* Cada un d'ells té un <*div*> intern amb la classe *.box-content* per posicionar la imatges de la icona amb un <*span*> i un paràgraf <*p*> per el text relacionat a cada icona. A més, el <*div*> amb la classe *.box-content* té un paràgraf <*p*> per el telèfon, el mail, i l'adreça respectivament.

4. El quart <*div*> té l'ID *.form-section* amb un <*article*> amb la classe *.box-form* i *.container* per ubicar el formulari. Aquest <*article*> conté un <*h4*> amb el títol, un paràgraf <*p*> per la descripció dels caràcter obligatoris, i el propi formulari <*form*> amb *method post* i la classe *.contact.*

   Dins el formulari primer trobem un <*div*> amb la classer *.form* que ens servirà per estructurar-lo en dues columnes. Cada una d'elles en un <*div*> amb la classe *.column*.

   - La primer columna conté tres <*div*> amb les classes *.name*, *.email* i *phone*. Cada un d'ells amb dos paràgrafs <*p*> que ens serveixen per col·locar primer un *label* i posteriorment el seu *input*, de tipus *text*, *email* i *tel* respectivament.

   - La segona columna conté tres <*div*> i un paràgraf <*p*>. Els <*div*> tenen les classes *.interest*, *.message* i *policy*. Seguint l'estructura de la primera columna, *.interest* conté un input de tipus *select* amb diferents *option*.* El <*div*> amb classe *.textarea* conté un <*textarea*> i el que té la classe *.policy* conté un *checkbox* per marcar la casella d'acceptació de la política de privacitat. El <p> té la classe *.submit.* per ubicar un <*button*>.

   *El primer dels *option* el fen *hidden* i amb el valor en blanc (i com a text * *) perquè no aparegui a la llista desplegable, ni com a valor per defecte ni per seleccionar.

   Cada label que conté l'asterisc (*) d'obligatorietat, està dins un <*abrr*> amb l'atribut *tittle="required".* Tots els elements obligatoris del formulari també tindran l'atribut *required* per garantir que es validin abans de ser enviats al servidor.

FOOTER (apareix a cada pàgina):

El *<footer>* té l'ID #footer que ens serveix per recollir l'enllaç de *skip-to-content, i* consta de de dos elements * *.

1. El primer té la classe *.column* i *.container*, que al mateix temps encapsula quatre * *:
    - El primer * * té la classe *.footer_logo*, i conté un *<h3>* amb el títol de la web.
    - El segon * * té la classe *.footer_menu* i conté un *<h3>* i un *<nav>* amb una llista no ordenada *<ul>* amb elements *<li>* encapsulant enllaços *<a>*, replicant el menú de la capçalera però de forma vertical.
    - El tercer * * té la classe *.footer_services* i conté un *<h3>* i un *<nav>* amb una llista no ordenada *<ul>* amb elements *<li>* encapsulant enllaços *<a>* per els serveis que ofereix la web.
    - El quart * * té la classe *.footer_social* i conté un *<h3>* i un *<nav>* amb una llista no ordenada *<ul>* amb elements *<li>* encapsulant enllaços *<a>* que contenen un *<img>* amb els logos de les diferents xarxes socials de l'empresa.

2. El segon té la classe *.credits* i *.container,* i conté un element * * i un paràgraf *<p>*.
    - El * * té la classe *.copyroght* i conté un paràgraf *<p>* per el text del copyright, i un altre * * amb la classe *.legal* per dos enllaços *<a>* de *Terms of Use* i *Provacy Policy*.

CSS

HEADER:

Fem que la capçalera *<header>* ocupi el 100% de l'espai de l'element pare (*l'article amb* la classe *.container*). Amb *'position:fixed;'* i *'top:0px;'* fem que sempre estigui visible a la part superior de la finestra del navegador. Amb el *z-index* ens assegurem que sempre quedi per sobre el contingut (li assignem 3 és per sobrepassar les capçaleres que veurem posteriorment a la taula en mode responsive):

```
header {
    position:fixed;
    top:0px;
    width: 100%;
    padding:15px 0px
    z-index: 3;
}
```

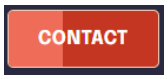
Dins del *<header>* tenim el *<nav>* que el fem *flex* per ubicar els elements interiors:

```
header nav {display: flex;}
```

Un dels elements del *<nav>* dins del *<header>* és la llista no ordenada *<ul>* que també fem *flex* per ubicar els diferents botons. En aquest cas li donem una amplada del 100%, alineem els elements al centre i els justifiquem amb *space-between*:

```
header nav ul {
    display:flex;
    align-items: center;
    justify-content: space-between;
    width: 100%;
    padding:0px;
    column-gap: 10px;
    list-style-type: none;
    margin:0;
}
```

L'enllaç *<a>* de 'Contact' té una animació per simular que, al passar per sobre, entri una barra per l'esquerra amb un color diferent al del fons original.



Per fer-ho ens ajudem del pseudoelement *:after* que és una capa que ocupa una alçada del 100%, però una amplada del 0% (per tant no es veu), amb un color diferent al fons original. L'ajustem a l'esquerra i dalt amb *'position:absolute;'*, *'top:0;'* i *'left:0;'* . Fem que l'efecte sigui lent amb *'transition: all .5s;'*.

```
 header nav ul.menu a.contact:after{
   position: absolute;
   content: "";
   top: 0;
   left: 0;
   width: 0;
   height: 100%;
   background: #ef6d58;
   transition: all .5s;
 }
```

L'enllaç té un ** amb el text, al que li assignem un *z-index* superior perquè la nova capa no tapi el text:

```
 header nav ul.menu a.contact span{
   position: relative;
   z-index: 1;
 }
```

Finalment, amb la combinació de la pseudoclasse *:hover* i el pseudoelement *:after (:hover:after)*, fem que al passar per sobre l'enllaç, la barra que tenia un 0% d'amplada, ara tingui un 100% i ocupi tot el fons. Gràcies al *transition* de la declaració anterior, l'efecte del canvi de color es produeix amb un cert retard i fa un efecte d'animació.

```
header nav ul.menu a.contact:hover:after{width: 100%;}
```

Aquest efecte només el volem amb la vista d'escriptori, i per tant les regles per *a.contact* les encapsulem dins un *mediaquery*:

```
@media (min-width: 769px)
```

MAIN (index.html):

El * * amb ID *#tittle* només té un *<h1>* i un paràgraf *<p>* que centrem:

```
#tittle {
    text-align:center;
    padding: 70px 0px;
}
```

(Aquest ID *#tittle* l'aprofitem també per *contact.html* i *color-theory-of-designers.html*)

En el primer *<section>* del * * amb l'ID *#experience* (que té la classe *.summary*) creem una estructura *grid* que ens permeti tenir dues columnes iguals. Una per el títol *<h2>* i l'altre per el * * que conté els dos paràgrafs *<p>*. A l'element *<h2>* li especifiquem que ocupi la primera columna, i que ocupi dues files. D'aquesta manera els dos paràgrafs *<p>* es col·loquen a la segona columna:

```
#index #experience .summary{
    margin: 50px 0px;
    display: grid;
    grid-gap: 10px;
    grid-template-columns: 50% 50%;
}
#index #experience .summary h2 {
    margin: 0px;
    font-size: 56px;
}
#index #experience .summary p{
    line-height: 2em;
}
```

En el * * dins de l'ID *#experience* que té la classe *.gallery* creem una estructura *grid* de 4 columnes amb un *gap* de 20px.

```
#index #experience .gallery {
    margin: 50px 0px;
    display:grid;
    grid-template-columns: repeat(4, 1fr);
    gap: 20px;
}
```

D'aquesta manera, podem fer que la primera imatge ocupi dues files i dues columnes:

```
#index #experience .gallery .redfabrics{
    grid-column: span 2;
    grid-row: span 2;
}
```

Posteriorment podem col·locar la resta d'imatges com volem, triant la fila i la columna on es posicionarà:

```
#index #experience .gallery .watchlamp{
    grid-column: 3;
    grid-row: 1;
}
#index #experience .gallery .papers{
    grid-column: 4;
    grid-row: 1;
}
#index #experience .gallery .laptop{
    grid-column: 3;
    grid-row: 2;
}
#index #experience .gallery .wallet{
    grid-column: 4;
    grid-row: 2;
}
```

Dins del *<section>* de l'ID *#experience* que té la classe *.blog-article*, tenim un altre * * amb la classe *.articles*, que també creem amb una estructura *grid* de 3 columnes amb un *gap* entre columnes de 20px:

```
#index #experience .blog-article .articles{
    display: grid;
    grid-template-columns: repeat(3, 1fr);
    column-gap: 20px;
}
```

Dins de *.blog-article* també hi ha un * * amb la classe *.btn-articles* que fem *flex* per poder centrar l'enllaç de 'More Articles':

```
#index #experience .blog-article .btn-articles {
    display: flex;
    justify-content: center;
    margin: 20px;
}
```

Dins del * * amb l'ID *#team* tenim un * * amb la classe *.team-members* amb una estructura *grid* de quatre columnes i un *gap* entre elles de 40px. Fem que tingui *'overflow:hidden;'* per amagar els membres de l'equip que per culpa de resolucions menors no es vegin sencers. Amb el valor auto, es generarà una barra d'*scroll* horitzontal per veure tots els membres. Especifiquem els colors de la barra d'scrol per *Firefox* i eliminem les fletxes de desplaçament:

```
#index #team .team-members {
    display: grid;
    grid-template-columns: repeat(4, 1fr);
    column-gap: 20px;
   /*estils d'scroll per Firefox*/
   scrollbar-color: #C43721 #ef6d58;
   scrollbar-width: thin;
   overflow: auto;

}
```

Dins del * * amb la classe *.member* trobem dos * * on el primer té la classe *.photo*, que fem *flex* per centrar-la. També li donem una alçada concret i fem que l'element intern (que serà la imatge) es col·loqui al final. D'aquesta manera quedarà espai entre la foto i la part superior d'aquest * *:

```
#index #team team-members .member .photo {
    display: flex;
    align-items: flex-end;
    justify-content: center;
    border: 1px solid #3a3c56;
    border-radius: 5px;
    width: 100%;
    height: 270px;
}
```

Donem estil a la barra d'*scroll* amb alçada, el border i colors tant de la barra de fons, com la de desplaçament:

```
#index #team .team-members::-webkit-scrollbar {height: 10px;}
#index #team .team-members::-webkit-scrollbar-track {
    background: #ef6d58;
    border-radius: 20px;
}
#index #team .team-members::-webkit-scrollbar-thumb {
    background-color: #C43721;
    border-radius: 20px;
}
```

* Com comentem al final de l'informe, a l'apartat 'Altres canvis', hem fet que el * * amb la classe *.team-members* tingui un *'overflow:hidden;'* a tot el document en comptes de nomes a la versió *responsive*, per fer que en resolucions petites (però no tant com per entrar dins del disseny *responsive*) també es generi un *scroll* horitzontal per els membres de l'equip.

El * * amb ID *#testimonials* té un *<h3>*, un *<h2>* i un paràgraf *<p>* que centrem:

```
#index #testimonials h3 {text-align: center;}
#index #testimonials h2 {
    text-align: center;
    font-size: 56px;
}
#index #testimonials .testimoni{
    margin: 0 auto;
    width: 70%;
    background-color: white;
    border-radius: 5px;
    color:#391400;
    padding:20px 50px;
    line-height: 2em;
}
```

El * * amb ID *#help* el fem *flex* per ubicar cada element en un costat amb un *gap* entre columnes de 60px:

```
#index #help {
    background-color: #ef6d58;
    display: flex;
    column-gap: 60px;
    padding: 50px 90px;
}
```

El * * *.info* el fem *flex,* per poder col·locar els elements uns sobre els altres amb *'direction:column;'*. Fem que ocupi la meitat de l'element pare:

```
#index #help .info {
    display: flex;
    flex-direction: column;
    width: 50%;
}
```

MAIN (color-theory-of-designers.html)

En el * * del *#tittle* del blog, li assignem un *background* amb la imatge i un *linear-gradient* amb el color blau principal, però amb una transparència. Transformem el color de RGB a hexadecimal per mantenir la coherència en tot el CSS, però el color RGB és *rgba(40, 41, 62, 0.7)*. En utilitzar els dos colors iguals en el *linear-gradient* aconseguim un mateix color pel fons. Tot i que a l'anunciat es demana una transparència del 20%, la fem d'un 70% per complir amb els estàndards d'accessibilitat pel que fa al contrast.

```
#blog #tittle {
    background: linear-gradient( #28293eb3, #28293eb3),
    url(../img/blog-header-bg.jpg);
}
```

L'apartat de on es parla de l'autor que té la classe *.summary-content* el fem grid i amb una proporció de 2fr 5fr, per col·locar la foto i el text. El text el col·loquem a la columna de la dreta dins un * * amb la classe *.summary-text*, que fem *flex* per poder posicionar els paràgrafs un al damunt de l'altre amb *'flex-direction: column;'*:

```
#blog #summary .summary-content {
    display: grid;
    grid-template-columns: 2fr 5fr;
    column-gap: 40px;
}
#blog #summary .summary-content .summary-text {
    padding: 20px;
    display: flex;
    flex-direction: column;
    align-self: center;
    line-height: 30px;
}
```

Totes les imatges de la pàgina estan dins d'un *<figure>* al que li anul·lem el *margin* per poder ajustar-les a l'espai que volem, i un *<figcaption>* amb el text del peu de pàgina, que reduïm una mica la mida del text i li afegim interlineat:

```
#blog figure {margin:0px}
#blog figure figcaption {
       font-size: 14px;
       line-height: 25px;
}
```

L'apartat del tex amb ID *#primmary-color.two-columns* el fem *flex* per col·locar les dues columnes interiors. A la primera que té la classe *.left-column* li posem un *padding* dret per separar-lo de les imatges que estan a la segona columna. La segona columna té la classe *.right-column* que fem *flex* i fem que el contingut vagi col·locant-se a les files inferior amb *'flex-wrap:wrap'*.

```
#blog #primarycolor .two-columns {
    display: flex;
    justify-content: space-between;
}

#blog #primarycolor .two-columns .left-column {padding-right: 150px;}

#blog #primarycolor .two-columns .right-column {
    display: flex;
    flex-wrap: wrap;
}
```

Per estilar els borders de la taula, ens interessa que només es vegin els interiors. També deixem sense border la separació de la capçalera i el cos de la taula. Per tant, primer assignem un color al *border* de tots els *<td>* i *<th>*, i després decidim quins amaguem.

De tots el *<th>* amaguem el *border* superior i inferior. També eliminem el border esquerra i dret del primer i últim *<th>* respectivament. Amb questes dos accions fem que la capçalera només tingui border en els costats dels *<th>* que no estiguin als extrems.

Pel cos de la taula, eliminem el border esquerra i dret del primer i últim *<td>* respectivament de cada *<tr>*. També eliminem el border superior i inferior de cada *<td>* del primer i últim *<tr>* respectivament. Així eliminem els borders externs laterals de cada fila, i els superior i inferior de la primera i última fila. Finalment eliminem el border esquerra del primer *<th>* dels *<tr>* dins del *<tbody>* i el border inferior del *<th>* de l'últim *<tr>* del *<tbody>*

Si en un futur la taula tingués més files o columnes, aquestes regles mantindrien els borders amb el disseny actual.

```
#blog #inbrief table th {
    border: solid 1px #c8c7c7;
    border-top: 0;
    border-bottom: 0;
}
#blog #inbrief table td {border: solid 1px #c8c7c7}
#blog #inbrief table th:first-child {border-left: 0;}
#blog #inbrief table th:last-child {border-right: 0;}
#blog #inbrief table td:first-child {border-left: 0;}
#blog #inbrief table td:last-child {border-right: 0;}
#blog #inbrief table tr:first-child td {border-top: 0;}
#blog #inbrief table tr:last-child td {border-bottom: 0;}
#blog #inbrief table tbody tr th:first-child{border-left: 0;}
#blog #inbrief table tbody tr:last-child th{border-bottom: 0;}
```

MAIN (contact.html):

Dins del * * amb ID *#contact-method* tenim el * * amb la classe *.contact-icons* que fem *grid* amb una estructura de 3 columnes de 1fr. Els elements interns d'aquest son els de cada mètode de contacte, que cada un d'ells té internament un * * amb la classe *.box-contact* que fem *flex* i en centrem el contingut.

```
#contact #contact-method .contact-icons {
    margin: 0 auto;
    width: 100%;
    color:#391400;
    border-radius: 5px;
    display: grid;
    grid-template-columns: repeat(3, 1fr);
    justify-content: center;
}
#contact #contact-method .contact-icons .box-contact{
    display:flex;
    align-items: center;
}
```

Dins de *.box-contact* hi ha un ** amb la classe *.contact-logo* que ens serveix per crear una caixa on anirà la imatge. El fem *flex* per centrar l'element *img* que contindrà i fem que sigui rodó amb el *border-radius*. Com que les imatges tenen mides diferents, es forcem a tenir la mateixa alçada i amplada.

```
#contact #contact-method .contact-icons .box-contact .contact-logo {
    display: flex;
    justify-content: center;
    width: 54px;
    height: 54px;
    background-color: #C43721;
    padding: 15px;
    border-radius: 50%;
}
#contact #contact-method .contact-icons .box-contact .contact-logo img{
    width: 25px;
    height: 25px;
}
```

A la part del formulari, en els elements *abbr* els hi eliminem els tres punts de la part inferior per deixar més net el text que fa mostrar l'asterisc del formulari:

```
#contact abbr{text-decoration: none;}
```

Fem que el * * que ens serveix per estructurar el formulari sigui *flex* amb un *gap* entre columnes de 50px. Alhora, els dos * * amb la classe *.column* els fem *flex* i *'flex-directio:column;'* per poder col·locar els elements interns del formulari un sobre l'altre a cada columna. Li donem un *width* del 50% perquè ocupin la meitat de l'espai disponible:

```
#contact #form-section .form{
    display: flex;
    column-gap: 50px;
    margin-top: 30px;
}
#contact #form-section .column{
    width: 50%;
    display: flex;
    flex-direction: column;
    row-gap: 20px;
}
```

FOOTER:

Fem que el primer * * del *<footer>* que té la classe *.column* sigui flex, i que cada un dels diferents * * que tenim a dins ocupi un 25% de l'amplada d'aquest contenidor.

```
footer .column {display:flex;}
footer .column div {
        width:25%;
        padding:40px;
}
```

Fem que la llista no ordenada *<ul>* per els icones de xarxes socials dins del * * amb la classe *.footer_social* sigui *flex* per posicionar cada element *<li>* intern que conté els icones:

```
footer .footer_social ul {display: flex;}
```

Cada icona de xarxa social és una imatge dins d'un enllaç *<a>* amb la classe *.social-icon*, que fem *flex* i el centrem tant vertical com horitzontalment al centre, amb una mida de 40x40px i amb els *border-radius* del 50% per fer el fons de la icona completament rodó.

```
footer .social-icon {
    display: flex;
    align-items: center;
    justify-content: center;
    width: 40px;
    height: 40px;
    margin: 0px 5px;
    border-radius: 50%;
    cursor: pointer;
    background-color: #C43721;
}
```

El * * del *<footer>* amb la classe *.copyright* és *flex* per col·locar el text del copyright i els enllaços dels textos legals un a cada extrem, amb *'jutify-content:space-between;':*

```
footer .copyright {
    display: flex;
    align-items: center;
    justify-content: space-between;
}
```

Al mateix temps, el * * amb classe *.legal* que conté els enllaços als textos legals és *flex* per tenir una estructura i separar tots dos enllaços amb *column-gap.*

```
footer .legal  {
    display:flex;
    column-gap: 60px;
}
```

Responsive:

Per fer el punt de trencament amb les resolucions de vista mòbil, fem una consulta als medis o, *mediaquey,* per resolucions menors de 768px:

```
@media (max-width: 768px)
```

Dins aquesta *mediaquery* apliquem tots els canvis en les regles que afecten a aquesta vista. El primer que fem és aplicar una mica de padding lateral a la classe *.container* ja que a diferència de la vista d'escriptori, tot el contingut quedava enganxat al lateral, i així millorem la visualització donant una mica d'espai:

```
.container {padding: 0 30px}
```

Capçalera responsive:

El primer dels canvis importants en el disseny responsiu el tenim a la capçalera i en el seu menú desplegable. Per aconseguir-ho fem que el botó del menú sigui un *label* relacionat amb l'ID i for a un *input* de tipus *checkbox* que ocultem amb *'display:none;'.* D'aquesta manera, quan fem *click* al botó del menú (*.menu-icon*), farem que l'*input* estigui *checked*, i a través de la pseudoclasse *checked* controlarem si el menú es veu, o no.

Dins del *<label>* creem un ** amb la classe *.nav-icon* per crear una de les barres del botó del menú. Fem que tingui una alçada de 2px i una amplada de 18px, i amb *'display:block;'* i *'position:relative;'* per posicionar-lo. El *transition* ens permetrà que quan desaparegui posteriorment ho faci de manera suau:

```
header .menu-icon .nav-icon{
    background: #fff;
    display: block;
    height: 2px;
    width: 18px;
    position: relative;
    transition: background .2s ease-out;
}
```

Posteriorment creem 2 barres més. Una amb el pseudoelement *::before* que tindrà un *'top:5px;'* per posar-la més amunt de la original, i l'altre amb *'top:-5px;'* per posicionar-la una mica per sota. Totes dues amb *'position:absolute;'* per col·locar-les a partir de l'element original. D'aquesta manera tenim el menú '*hamburguesa'* per la seva similitud visual:

```
header .menu-icon .nav-icon:before{
    background: #fff;
    content: "";
    display: block;
    height: 100%;
    width: 100%;
    position: absolute;
    transition: all .2s ease-out;
    top:5px;
}
header .menu-icon .nav-icon:after{
    background: #fff;
    content: "";
    display:block;
    height: 100%;
    width: 100%;
    position: absolute;
    transition: all .2s ease-out;
    top:-5px;
}
```

Per fer que el menú d'hamburguesa canviï a una creu quan fem *click*, utilitzem la pseudoclasse *:checked* de la classe *.menu-btn.* Quan tinguem *.menu-btn:checked* farem que *.menu* tingui una alçada màxima de 700 px que serà suficient per que es vegin tots els elements del menú. També farem que la barra central del botó (*.nav-icon*) desaparegui, i que la barra superior (*.nav-icon:before*) i la inferior (*.nav-icon:after*) canviïn el seu angle:

```
header .menu-btn:checked ~ .menu {max-height: 700px;}
header .menu-btn:checked ~ .menu-icon .nav-icon{background-color: transparent;}
header .menu-btn:checked ~ .menu-icon .nav-icon::before{
    transform: rotate(-45deg);
    top:0
}
header .menu-btn:checked ~ .menu-icon .nav-icon:after{
    transform: rotate(45deg);
    top:0;
}
```

* Ens ajudem del combinador ~ per indicar que volem afectar les classes que son germanes i depenen del mateix pare, ja que tant *.menu* com *.menu-icon* son elements situats al mateix nivell que *.menu-btn*.

A nivell de visualització del menú, el primer que fem és que el contenidor *<nav>* i la llista no ordenada *<ul>* perdin la seva propietat de flex, amb *'display:revert;'*. D'aquesta manera perden la seva disposició en columnes, i es tornen a col·locar segons el flux normal del document:

```
header nav {display:revert;}
header nav ul {
    display: revert;
    overflow: hidden;
}
```

* *'overflow:hidden;'* del *<ul>* ens serveix per que no es vegin els elements del menú quan aquesta estigui replegat.

Fem que els enllaços *<a>* del menú ocupin tot l'ample de la vista amb '*display:block;*' i alineem el text al centre:

```
header nav ul.menu a {
    display: block;
    padding: 20px;
    font-weight: bold;
    font-size: 50px;
    text-align: center;
    border-bottom: 5px solid #28293e;
}
```

Fem que el logo quedo a l'esquerra i per sobre el menu amb *'float:left;'* i *'display:block;'*:

```
header .logo {
    float: left;
    display: block;
    padding: 15px 10px;
}
```

Per defecte fem que el menú quedi ocult amb '*max-height:0;'*, i amb el *transition* farem que quan el mostrem, aparegui amb suavitat i no de cop.

```
header .menu {
    max-height: 0;
    transition: max-height .2s ease-out;
}
```

* Com hem vist abans, quan premem el botó del menú i tinguem la classe *.menu-btn:checked*, el menú tindrà un *max-height* de 700px que ens mostrarà els elements del menú.

Índex responsive:

A l'índex fem que el * * amb la classe *.summary* perdi el grid-template-columns anterior del 50% 50%. D'aquesta manera els elements es col·loquen uns sobre els altres:

```
#index #experience .summary {grid-template-columns: revert;}
```

Per modificar la galeria, fem que cada element ocupi els espais i les columnes diferents tal com es mostra a la imatge de mostra:

```
#index #experience .gallery .redfabrics {
    grid-column: span 4;
    grid-row: span 4;
}
#index #experience .gallery .watchlamp {
    grid-row: span 2;
    grid-column: span 2;

}
#index #experience .gallery .papers {
    grid-row: span 2;
    grid-column: span 2;
}
#index #experience .gallery .laptop {
    grid-row: span 2;
    grid-column: span 2;
}
#index #experience .gallery .wallet {
    grid-row: span 2;
    grid-column: span 2;
}
```

Fem que la caixa que conté els articles del blog perdi el *grid-template-columns*, i d'aquesta manera es col·loquin un sobre els altres en comptes de 3fr.

```
#index #experience .blog-article .articles {grid-template-columns: revert;}
```

El * * amb la classe *.btn-articles* el fem *flex* amb i alineació a l'esquerra, i així podem posicionar el botó a l'esquerra:

```
#index #experience .blog-article .btn-articles {
    display: flex;
    justify-content: left;
    margin: initial;
}
```

Pel bloc de text del testimoni volem que en comptes de ser una caixa amb el text, sigui un bloc que ocupi tota l'amplada de la pagina. Per fer-ho, fem que el * * amb l'ID *#testimoni* tingui tota l'amplada, i li traiem el *border-radius*:

```
#index #testimoni{
    width: 100%;
    background-color: white;
    border-radius: 0px;
    padding: 20px;
}
```

Per l'últim apartat de l'índex tenim el * * amb ID *#help*. Al div interior amb la classe *.helpbox* li fem el *'flex-direction:colom-reverse;'* per tal de col·locar les dues columnes en una de sola, i que la segona es posicioni en primera posició. També fem que la primera columna (classe *.info*) passi a ocupar el 100% de l'amplada:

```
#index #help .helpbox {flex-direction: column-reverse;}
#index #help .helpbox .info {width: 100%;}
```

Blog responsive:

El primer que fem en el blog, és eliminar revertir el *grid-template-columns* del div amb classe *.summary-content* per fer que els elements es col·loquin segons el flux normal, i perdin les proporcions de dues columnes amb la relació 2fr 5fr. També eliminem el *padding* per tal de guanyar amplada del text:

```
#blog #summary {padding: revert;}
#blog #summary .summary-content {grid-template-columns: revert;}
#blog #summary .summary-content .summary-text {padding:0;}
```

Fem que la imatge ocupi la meitat de la seva amplada original, i la fem pujar per sobre dels límits del seu element pare amb un *margin-top* negatiu, per tal de que quedi com a la imatge de mostra:

```
#blog #summary .summary-content .summary-author figure img {width: 50%;}
#blog #summary .summary-content .summary-author figure  {margin-top: -100px}
```

La part central del blog la convertim en una sola columna per text i imatges, modificant *flex-direction* del * * amb la classe *.two-columns*:

```
#blog #primarycolor .two-columns {flex-direction: column;}
```

La taula també és un altre dels canvis importants en el disseny responsive. Per tal d'adaptar-la a la vista mòbil com demana la imatge d'exemple, el primer que fem és ocultar la primera columna, sencera, i que tal com havíem dissenyat la taula, son tots els primers *<th>* de cada fila *<tr>*:

```
#blog #inbrief #table-inbrief tr th:first-child {
    position: absolute;
    visibility: hidden;
}
```

Posteriorment, creem una nova capa que farem amb el pseudoelement *::before* amb el contingut del *<th>* que hem eliminat de cada fila de contingut dins del *<tbody>* (i per tant, excloent la capçalera principal del *<thead>*). Ho fem amb la propietat content, i el valor de *attr(data-label)*, que prèviament hem col·locat al document html amb el text que volem. A més fem que aquesta capa (*td:before*) tingui un *'position:absolut;'* per poder posicionar-la dins el *<td>* amb *left* i *top* a 0. Fem que tingui un *width* del 180% perquè volem que sobresurti per la dreta i ocupi part del *<td>* de la segona columna (també en dispositius en baixes resolucions):

```
#blog #inbrief #table-inbrief td:before {
    content: attr(data-label);
    position: absolute;
    left: 0;
    top: 0;
    width: 180%;
    z-index:2;
    color:#fff;
    background-color: #989494;
    font-weight: bold;
    padding:20px 40px;
}
```

* el *'z-indez:2;'* fa que quedi per sobre de la capa que genera el :*before* següent.

Després fem la mateixa capa que abans, però a l'últim *<td>* fem que el content sigui un espai en blanc, i que ocupi el 100% de l'amplada amb '*z-index:1;'* perquè es col·loqui per sota del *:before* anterior i no tapi el text. Això ens serveix per simular que el *:before* anterior ocupa l'ampla total de la taula.

```
#blog #inbrief #table-inbrief td:last-child:before{
    border-left: 1px solid #c8c7c7;
    content: "\00a0";
    width: 100%;
    z-index: 1;
}
```

* si la taula hagués de créixer en un futur amb més columnes (més *<td>* en el model responsive) aquesta regla la podríem canviar de *'td:last-child:before'* a *'td:not(:nth-child(2)):before'* per fer que qualsevol *<td>* excepte el segon (tenint en compte que :nth-child compta el *<th>* com si fos el primer *<td>).* D'aquesta manera qualsevol columna que no fos la que conté el text que generem amb el *content* tindria aquesta estil.

NOTA: Quan he volgut provar la web a un mòbil real, la taula de l'apartat del blog sobrepassava el límit lateral de la pantalla, i feia que existís un scroll horitzontal a tota la web, ja que tant la taula com el *<header>* eren més amples que la pantalla. Per corregir-ho, tot i que a l'anunciat no s'especificava, he fet un nou *mediaqury* per resolucions mes petites, on redueixo el *padding* de les cel·les per tal de que la taula no sigui tant ample i evitar el problema *d'scroll* horitzontal:

```
@media (max-width: 465px) {
    #blog #inbrief #table-inbrief td {padding: 20px 10px 10px;}
    #blog #inbrief #table-inbrief tr:first-child th,
    #blog #inbrief #table-inbrief td:before {padding: 10px 10px;}
}
```

* el problema torna a aparèixer en resolucions menors de 333px, però considero que és massa petita per seguir fent responsiu un disseny en aquestes resolucions.

Contacte responsive:

Fem que les caixes dels icones dels diferents tipus de mètodes de contacte es posicionin en una sola columna amb *'flex-direction: column;'*:

```
#contact #contact-method .contact-icons {flex-direction: column;}
```

Modifiquem els borders ja que al estar en columna es modifiquen els de la vista d'escriptori. A la primera caixa li amaguem el border inferior, i modifiquem les cantonades perquè siguin arrodonides les de la part superior. De la tercera caixa amaguem el border superior i fem que només tingui arrodonides les cantonades inferiors :

```
#contact #contact-method .contact-icons .phone {
    border-radius: 6px 6px 0px 0px;
    border-bottom: hidden;
    border-right: 1px solid #efd4c6;
}
#contact #contact-method .contact-icons .address {
    border-radius: 0px 0px 6px 6px;
    border-top: hidden;
    border-left: 1px solid #efd4c6;
}
```

Fem que el formulari pasi a tenir una sola columna, fent que el * * amb la classe *.form* tingui un *'flex-direction:column;'* i li afegim un *row-gap* per espaiar els dos blocs:

```
#contact #form-section .form {
    flex-direction:column;
    row-gap: 20px;
}
```

Fem que el * * amb la classe *.column* tingui un 100% d'amplada per tal que tots els elements ocupin el màxim possible. Alineem el *p.submit* al centre per col·locar el botó en aquesta posició:

```
#contact #form-section .column {width: 100%;}
#contact #form-section p.submit {text-align: center;}
```

Footer responsive:

Fem que tant el * * amb la classe *.column* com el de la classe *.copyright* passin a tenir un *'flex-direction:column;'* i una amplada del 100%. Centrem tots els elements i els donem una mica d'espai per millorar-ne la visualització:

```
footer .column {
    flex-direction: column;
    align-items: center;
    text-align: center;
    row-gap: 20px;
}
footer .column div {width: 100%;}
footer .footer-social ul {justify-content: center;}
footer .copyright {
    flex-direction: column;
    row-gap: 20px;
}
footer p.freepik-credits {
    text-align: center;
    padding-top: 20px;
}
```

Canvis per millorar l'accessibilitat:

Canviem el color dels h3 i del ** amb classe *.agency* de color #C43721 al #F8715B per complir el nivell de contrast de 4.5.

Canviem la opacitat de la transparència pel títol del Blog de 20% a 70% per complir el nivell de contrast de 4.5.

A la versió responsive, fem que quan el *checkbox* amb la classe *.menu-btn* que tenim ocult, tingui el *focus*, el botó que té la classe *.menu-icon* tingui l'estil del *focus*, i quan el perdi l'estil desaparegui. També fem que per defecte els elements del menú no apareguin, i només ho facin quan tenim *.menu-btn:checked.* Amb això fem que si el menú no està desplegat, si naveguem per teclat, el *focus* passa del botó d'hamburguesa al contingut web i no per els enllaços ocults:

```
    header .menu-btn:focus ~ .menu-icon {outline: 3px dashed #C43721;}
    header .menu-btn:not(:focus-visible) ~ .menu-icon {outline: none;}
    header .menu a {visibility: hidden; }
    header .menu-btn:checked ~ .menu a {visibility: visible;}
```

* al tenir el *label* enllaçat amb un *checkbox*, no he trobat una forma que no sigui amb *JavaScript* per fer que el menú d'hamburguesa sigui interactiu amb la tecla enter, i per tant ho és amb l'espai.

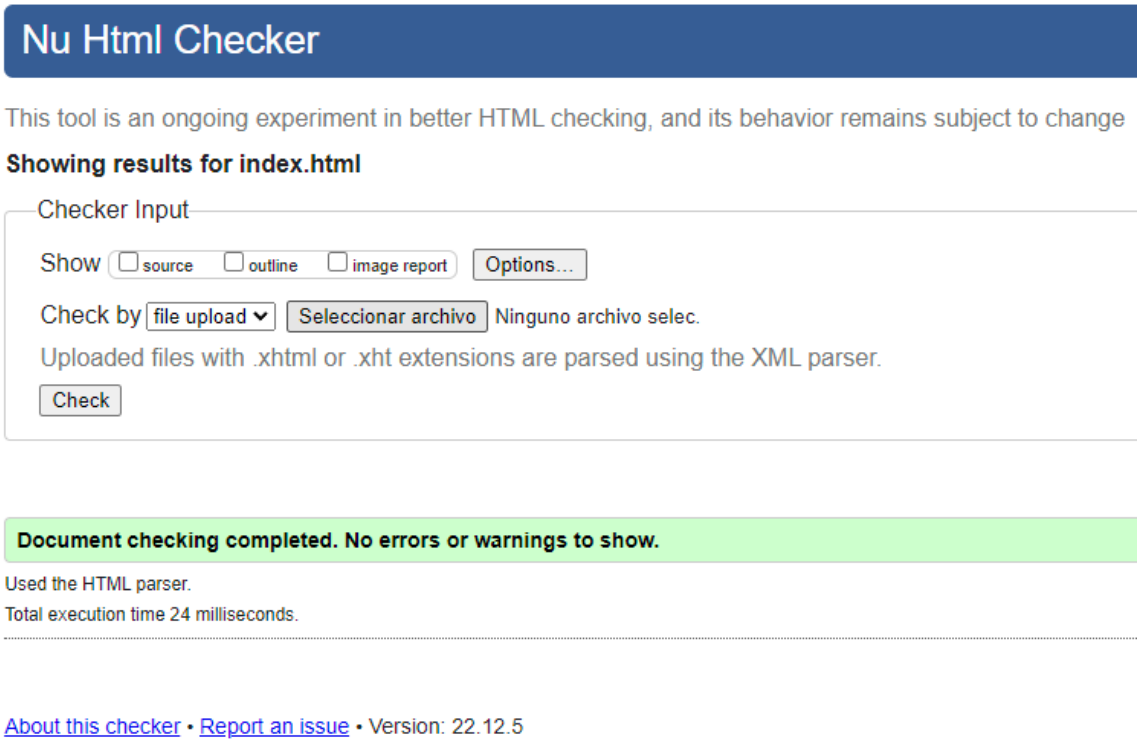
Altres canvis:

Reduïm la mida de lletra de *<table>* de 24px a 20px.

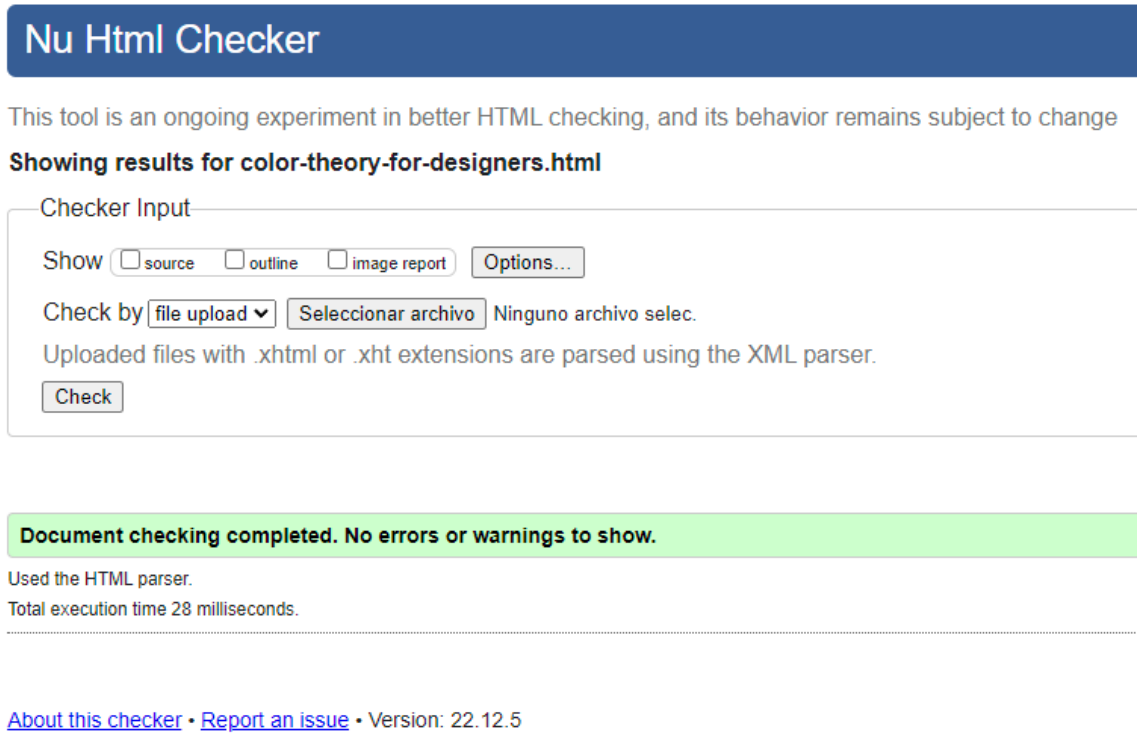
Per els membres de l'equip teníem inicialment el *'overflow:hidden'* només per la versió responsive, però per evitar que els membres desapareguin a resolucions pròximes al disseny *responsive*, hem aplicat el *'overflow:hidden'* a tot el document.

Validació de fitxers:

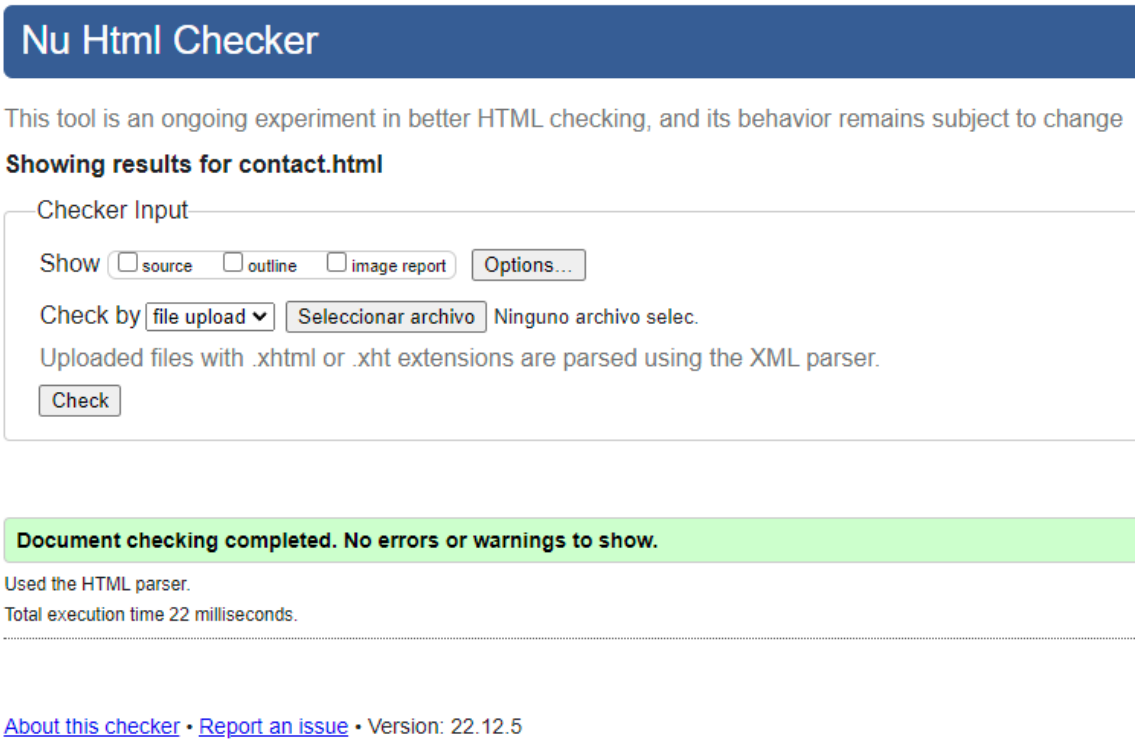
index.html



color-theory-for-designers.html



contact.html



style.css