Transitions, animations, transformations

(ex : création d'un diaporama)

Rédigé par: Jean-Luc Colson			
Date première rédaction: Sep 07			
SUIVI DES MODIFICATIONS A LA FICHE			
Série	Date	Page(s) modifiée(s)	Raison

Contents

Exercice : créer un diaporama en HTML et CSS

L'objectif de ce nouvel exercice est d'essayer de créer un diaporama, c'est-à-dire un enchainement fluide d'images, en HTML et en CSS.

L'idée ici va être d'utiliser judicieusement la propriété background-image et les animations et transformations pour modifier l'image de fond d'un élément et ainsi créer un effet de diaporama.

Nous n'allons évidemment pas pouvoir coder certaines fonctionnalités que possèdent les diaporamas « complets » créés avec du JavaScript mais allons essayer de réaliser quelque chose qui va s'en rapprocher.

Je vous propose ici de créer deux animations différentes pour créer deux diaporamas : une première animation de fondu et une seconde qui va faire défiler nos images comme dans le cas d'un diaporama classique.

De plus, pour chaque animation de diaporama, nous allons créer deux versions : une version avec un diaporama de taille fixe et une version avec un diaporama qui va s'adapter en fonction de la taille de la fenêtre.

Création d'un diaporama avec effet de fondu

Pour notre premier diaporama, nous allons vouloir créer un effet de fondu, c'est-à-dire de disparition progressive d'une image et d'apparition progressive d'une autre.

Pour cela, je vous propose de travailler en 3 étapes :

1. Création des cadres du diaporama en HTML ;
2. Mise en forme des cadres en CSS ;
3. Création de l'animation en CSS.

Le code HTML du diaporama

A manière la plus simple de créer un diaporama en HTML et en CSS va être de faire défiler des images de fond.

Pour pouvoir faire cela, il va nous falloir un élément conteneur ou cadre auquel on va ensuite pouvoir passer des images de fond.

Nous allons ainsi nous contenter en HTML d'utiliser des éléments div comme cadres.

On va créer un premier div class="d1" qui va représenter notre cadre de diaporama à dimension fixe et un deuxième div class="d2" qu'on va lui-même placer dans un div class="conteneur" et qui va représenter notre cadre de diaporama aux dimensions adaptables.

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Cours HTML et CSS</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="cours.css">
    </head>

    <body>
        <div class="d1"></div>
        <div class="conteneur">
            <div class="d2"></div>
        </div>
    </body>
</html>
```

Mise en forme des cadres en CSS

Pour créer nos diaporamas en CSS nous allons nous servir de la propriété background-image que nous allons ensuite animer grâce à la propriété animation.

En pratique, la propriété background-image s'utilise souvent pour apporter un fond à un élément HTML qui possède du contenu et donc une hauteur définie.

Ici, nos div qui servent de cadres à notre diaporama ne vont contenir que des images de fond et aucun « vrai » contenu. Par défaut, leur hauteur va donc être nulle puisqu'une image de fond n'est qu'un fond par définition et qu'un fond ne peut pas impacter la taille d'un élément.

Nous allons donc devoir préciser explicitement une hauteur pour nos conteneurs. Nous allons pouvoir le faire de deux façons différentes selon qu'on souhaite créer un diaporama avec un cadre possédant des dimensions fixes ou en pourcentage.

Cadre de diaporama avec dimensions fixes

Pour utiliser un cadre de diaporama avec dimensions fixes, nous allons tout simplement préciser une largeur et une hauteur explicites en px pour notre div.

Il faudra ensuite recadrer nos images de fond à la même taille que le div ou à minima faire en sorte qu'elles possèdent le même ratio largeur / hauteur que celui-ci afin d'avoir un bon affichage.

L'avantage de cette première méthode est qu'on va avoir un comportement stable et prévisible pour les dimensions de notre diaporama. En contrepartie, le diaporama fera toujours la même taille quelle que soit la taille de l'écran de vos visiteurs.

Notez qu'on va pouvoir définir plusieurs tailles fixes selon certains paliers de tailles grâce aux media queries que nous étudierons dans la suite de ce cours.

```css
.d1{
    width: 576px;
    height: 432px;
    margin: 50px auto;
    box-shadow: 0px 15px 10px -5px #777;
    background-color: #EDEDED;
    background-size: contain;
}
```

Ici, on indique explicitement des dimensions pour notre div qui va contenir notre diaporama par défaut ainsi qu'une couleur de fond. Pour le moment, nous créons simplement le cadre et n'utilisons donc pas d'image.

Il est toutefois toujours essentiel de préciser une couleur de fond au cas où les images du diaporama ne pourraient pas pour une raison ou une autre s'afficher.

Ensuite, on définit un background-size: contain pour que nos images de fond soient à la fois contraintes dans le conteneur mais occupe le plus d'espace dans celui-ci tout en conservant leur proportions d'origine.

On crée également un effet d'ombre sous notre div avec box-shadow pour donner l'impression qu'il est au-dessus de la page.

Cadre de diaporama avec des dimensions en %

Créer un cadre de diaporama qui va se redimensionner en même temps qu'on va changer la taille de la fenêtre va être un peu plus complexe.

En effet, je vous rappelle qu'une image de fond n'est pas considérée comme un contenu en soi et donc que la hauteur de notre cadre est nulle par défaut.

Ici, quand on agrandit ou rétrécit la fenêtre, il va falloir que la taille de notre cadre s'adapte et que nos images de fond s'affichent entièrement dans tous les cas.

Pour faire cela, nous allons utiliser un petit hack CSS. Nous allons définir explicitement une hauteur nulle pour notre div et une largeur égale à 100% et utiliser la propriété padding-top.

Le padding-top va ici servir à donner une hauteur au cadre. On va lui passer une valeur en pourcentage qui va correspondre au ratio hauteur / largeur de l'image qu'on souhaite voir s'afficher.

Par exemple, si mon image de fond fait 1200px de large par 900px de haut, le ratio hauteur / largeur est de 900/1200 = 3/4 = 75%. On indiquera donc un padding-top: 75% dans ce cas.

Cela va faire que notre cadre va se redimensionner en gardant toujours ce ratio de ¾, c'est-à-dire qu'il aura toujours une hauteur égale à 75% de sa largeur.

```css
.d2{
    width: 100%;
    height: 0px;
    padding-top: 75%;
    margin: 50px auto;
    box-shadow: 0px 0px 10px #777;
    background-color: #EDEDED;
    background-size: contain;
}
```

Ici, d'un point de vue purement esthétique, on fait cette fois-ci le choix d'utiliser un box-shadow centré autour du div.

L'avantage de cette deuxième méthode est que notre cadre de diaporama va pouvoir se redimensionner en même temps que la fenêtre. L'inconvénient est qu'on ne va pas pouvoir maitriser la hauteur du diaporama qui va pouvoir atteindre une très grande taille sur de grands écrans, ce qui peut être un comportement indésirable.

C'est la raison pour laquelle nous avons placé notre div class="d2" dans un autre div class="conteneur" qui va nous servir de conteneur et pour lequel on va préciser une taille maximale avec max-width afin de limiter la taille de notre diaporama à partir d'une certaine taille de fenêtre.

```css
.d2{
    width: 100%;
    height: 0px;
    padding-top: 75%;
    box-shadow: 0px 0px 10px #777;
    background-color: #EDEDED;
    background-size: contain;
}
```

```css
.conteneur{
    max-width: 576px;
    margin: 50px auto;
}
```

Création de l'animation de fondu du diaporama

Il ne nous reste plus qu'à créer notre animation de fondu pour rendre notre diaporama fonctionnel. Pour ce diaporama, je vais utiliser trois images que vous pouvez trouver dans le répertoire images. La quatrième image dans le dossier va être utilisée pour notre prochain diaporama.

L'animation de fondu va être relativement simple à réaliser : nous allons simplement passer nos différentes images à la propriété background-image à différents moments de l'animation.

Comme le principe d'une animation est de passer progressivement d'une valeur de départ à une valeur d'arrivée, nos images vont s'enchainer avec un effet de fondu de façon naturelle.

```css
@keyframes fondu{
    0%{background-image: url("diapo1.png");}
    33.33%{background-image: url("diapo2.png");}
    66.67%{background-image: url("diapo3.png");}
    100%{background-image: url("diapo1.png");}
}
```

Nous allons répéter l'animation à l'infini pour que nos images s'enchainent constamment et allons préciser le comportement du background-image avant et après l'animation au cas où celle-ci ne puisse pas se lancer.

```css
.d1{
    width: 576px;
    height: 432px;
    margin: 50px auto;
    box-shadow: 0px 15px 10px -5px #777;
    background-color: #EDEDED;
    background-size: contain;
    animation: fondu 15s ease-in-out infinite both;
}
.conteneur{
    max-width: 576px;
    margin: 50px auto;
}
.d2{
    width: 100%;
    height: 0px;
    padding-top: 75%;
    box-shadow: 0px 0px 10px #777;
    background-color: #EDEDED;
    background-size: contain;
    animation: fondu 15s ease-in-out infinite both;
}
```

Finalement, nous allons vouloir mettre en pause l'animation lorsqu'un utilisateur passe sa souris sur le cadre du diaporama, pour lui laisser le temps d'apprécier l'image.

```css
.d1:hover, .d2:hover{
    animation-play-state: paused;
}
```

Voilà tout pour notre premier effet de diaporama. Vous pouvez retrouver le code complet ci-dessous:

D:\Data\Cours\ISL\HTML CSS\MonSite\html\transitions-30 diaporama.html - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

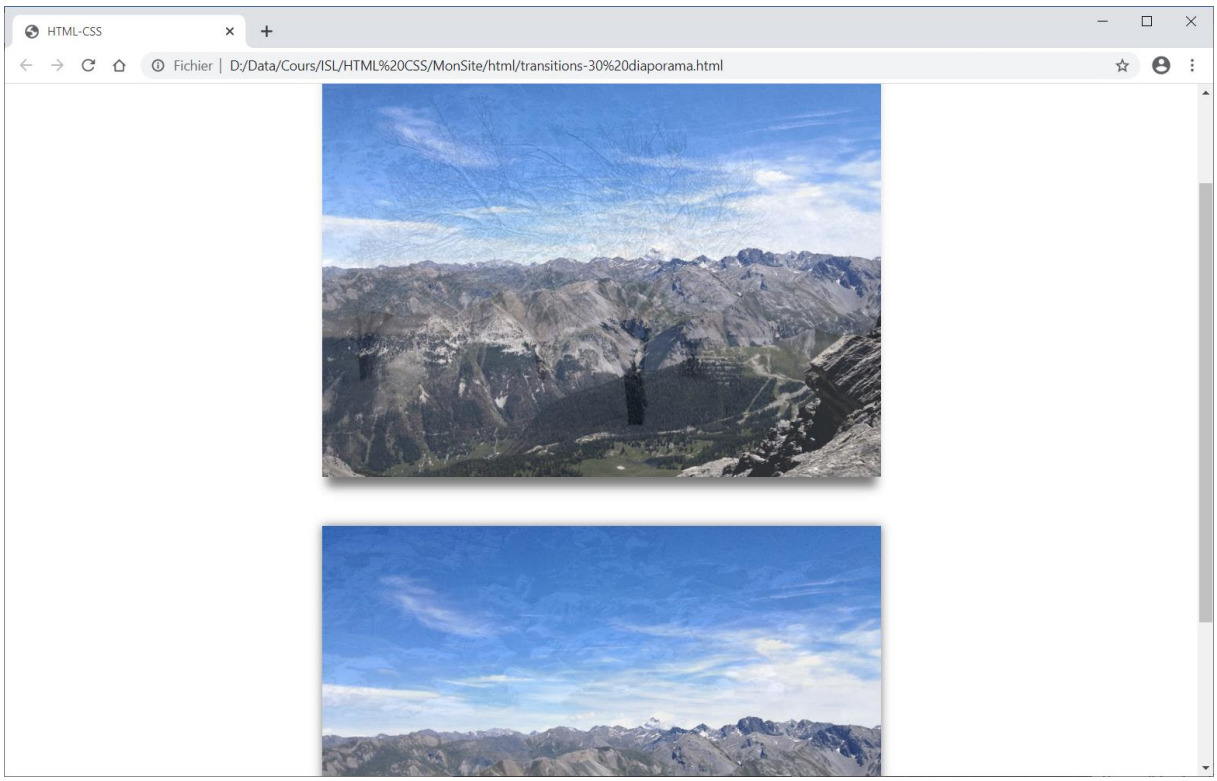
transitions-31 diaporamafinal.html | transitions-30 diaporama.html

```
<!DOCTYPE html>
<html>
<head>
    <title>HTML-CSS</title>
    <meta charset="utf-8">
    <link rel="stylesheet"
    href="..\css\transitions-30 diaporama.css">
</head>
<body>

<h1>HTML-CSS</h1>

<div class="d1"></div>
<div class="conteneur">
<div class="d2"></div>
</div>
</body>
</html>
```

transitions-31 diaporamafinal.css | transitions-30 diaporama.css

```
.d1{
    width: 576px;
    height: 432px;
    margin: 50px auto;
    box-shadow: 0px 15px 10px -5px #777;
    background-color: #EDEDED;
    background-size: contain;
    animation: fondu 15s ease-in-out infinite both;
}
.conteneur{
    max-width: 576px;
    margin: 50px auto;
}
.d2{
    width: 100%;
    height: 0px;
    padding-top: 75%;
    box-shadow: 0px 0px 10px #777;
    background-color: #EDEDED;
    background-size: contain;
    animation: fondu 15s ease-in-out infinite both;
}
.d1:hover, .d2:hover{
    animation-play-state: paused;
}
@keyframes fondu{
    0%{background-image: url("../images/diapo1.png");}
    33.33%{background-image: url("../images/diapo2.png");}
    66.67%{background-image: url("../images/diapo3.png");}
    100%{background-image: url("../images/diapo1.png");}
}

```

Hyper Text Markup Language file | length : 293 lines : 18 | Ln : 14 Col : 24 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS



Création d'un diaporama avec défilement des images

Essayons maintenant de créer un diaporama avec un effet de défilement des images en fond.

L'idée ici va être de créer une maxi-image qui va contenir toutes les images de notre diaporama. Nous allons ensuite placer cette image en fond et allons animer sa position pour la faire dérouler.

Pour animer la position de l'image de fond, nous allons pouvoir soit utiliser la propriété background-position soit la propriété transform avec sa fonction translate(). Cette deuxième méthode est à privilégier car elle est plus performante.

Code HTML du diaporama avec effet de défilement

Nous allons à nouveau essayer de créer deux diaporamas : un avec des dimensions fixes et un qui va s'adapter en fonction de la taille de la fenêtre.

Nous allons donc déjà avoir besoin de deux cadres pour nos diaporamas qui vont être représentés par deux div. Ici, nous allons placer chacun de ces cadres dans un autre div conteneur.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Cours HTML et CSS</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="cours.css">
    </head>

    <body>
        <div class="conteneur1">
            <div class="d1"></div>
        </div>
        <div class="conteneur2">
            <div class="d2"></div>
        </div>
    </body>
</html>
```

Diaporama de dimensions fixes avec effet de défilement

Commençons déjà par essayer de créer un diaporama avec effet de défilement et des dimensions fixes.

L'idée derrière l'effet de défilement va être ici de n'utiliser qu'une grande image qui va être en fait composée de plusieurs images et de la faire bouger.

On peut donc déjà définir notre image de fond avec background-image. Je vais pour ma part utiliser une image qui a été créée à partir des 3 images du diaporama précédent et que j'ai appelée diapo123.png.

Mon image fait ici 2880px de largeur par 720px de hauteur et est composée de 3 images de tailles identiques (960px*720px).

On va vouloir que l'effet de défilement soit infini et nous allons créer cet effet avec une transformation de type translation.

On va déjà commencer par donner une taille égale à notre grande image + la largeur d'une sous image à notre div afin de pouvoir créer une boucle infini fluide. Dans mon cas, il faut donc que je donne une taille de 2880px + 960px = 3840px à mon div. Nous allons fixer une hauteur du div égale à celle de l'image de fond à savoir 720px.

```
.d1{
    width: 3840px;
    height: 720px;
    background-color: #EDEDED;
    background: url("diapo123.png");
    background-size: contain;
}
```

Nous avons donc un cadre de 3840px de largeur. Cependant, nous voulons que la partie visible du diaporama soit égale à la largeur de chacune de nos sous images (on pourrait tout à fait décider d'une largeur différente mais c'est la largeur qui fait le plus de sens selon moi).

Pour cela, nous allons passer une largeur maximale à notre div conteneur égale à la largeur de nos sous images et également lui passer un overflow : hidden pour cacher ce qui dépasse du conteneur.

```
.conteneur1{
    overflow: hidden;
    max-width: 960px;
    margin: 50px auto;
    box-shadow: 0px 15px 10px -5px #777;
}
```

Finalement, nous allons créer notre effet de défilement en animant une transformation de type translation. Au début de notre animation, on va vouloir afficher la première sous image dans le cadre puis on va vouloir faire défiler la grande image jusqu'à arriver à nouveau sur une vue montrant la première sous image.

```
@keyframes defilement1{
    0%{transform: translate(0,0);}
    100%{transform: translate(-2880px,0);}
}
```

A la fin de notre animation, l'image s'est déplacée de sa taille exactement de sa largeur. On utilise ici la fait qu'une image de fond est par défaut répétée pour remplir le fond d'un élément, ce qui fait que notre image de fin d'animation est la même que celle du début (la première sous image est bien répétée du fait que la largeur du cadre soit égale à celle de l'image de fond + la largeur d'une sous-image).

Ensuite, il ne nous reste plus qu'à répéter cette boucle à l'infini en définissant un nombre d'animations infini.

```css
.d1{
    width: 3840px;
    height: 720px;
    background-color: #EDEDED;
    background: url("diapo123.png");
    background-size: contain;
    animation: defilement1 12s linear infinite;
}
```

Diaporama avec effet de défilement aux dimensions adaptables

Nous allons finalement pouvoir créer un diaporama avec effet de défilement et aux dimensions adaptables sur le même modèle que ce qu'on a pu faire précédemment mais en convertissant les différentes valeurs en pourcentage.

La principale difficulté/ astuce ici va être de jouer avec le ratio de notre image et la largeur de notre cadre pour faire en sorte que les sous images s'affichent complètement à chaque fois et que le défilement soit fluide.

Pour rappel, notre image de fond fait 2880px*720px ce qui signifie qu'elle est quatre fois plus large que haute. Chacune des sous images qu'elle contient à un ratio largeur/ hauteur de 4 : 3.

Ici, on va commencer par passer une largeur width : 400% à notre cadre. Ensuite, on va lui attribuer un padding-top : 75% afin que la hauteur du div soit toujours égale à 75% de la largeur visible du div.

La partie visible du div cadre aura donc toujours un ratio de 4 : 3. En utilisant background-size : contain, la première répétions de notre maxi image de fond va donc prendre une largeur égale à 300% de la partie visible du div.

Comme notre div cadre possède une largeur de 400%, notre image de fond va donc s'afficher une fois complètement dedans puis un tiers de l'image va se répéter (ce qui va correspondre à notre première sous image se répétant).

```css
.d2{
    width: 400%;
    height: 0;
    padding-top: 75%;
    background-color: #EDEDED;
    background-image: url("diapo123.png");
    background-size: contain;
}
```

On va ensuite passer une largeur maximale à notre div conteneur qu'on va définir ici comme égale à une de nos sous images pour éviter que notre diaporama ne dépasse une certaine taille sur les grands écrans. On passe également un overflow : hidden pour cacher la partie du diaporama qui dépasse de l'écran.

```css
.conteneur2{
    max-width: 960px;
    overflow: hidden;
    margin: 50px auto;
    box-shadow: 0px 0px 10px #777;
}
```

Il ne nous reste plus qu'à définir notre @keyframes avec notre translation et les propriétés de notre animation en soi.

De manière similaire à ce qu'on a pu faire précédemment, on va faire en sorte que la première boucle de l'animation des termine exactement lorsque notre première sous image occupe à nouveau l'espace visible dans le cadre.

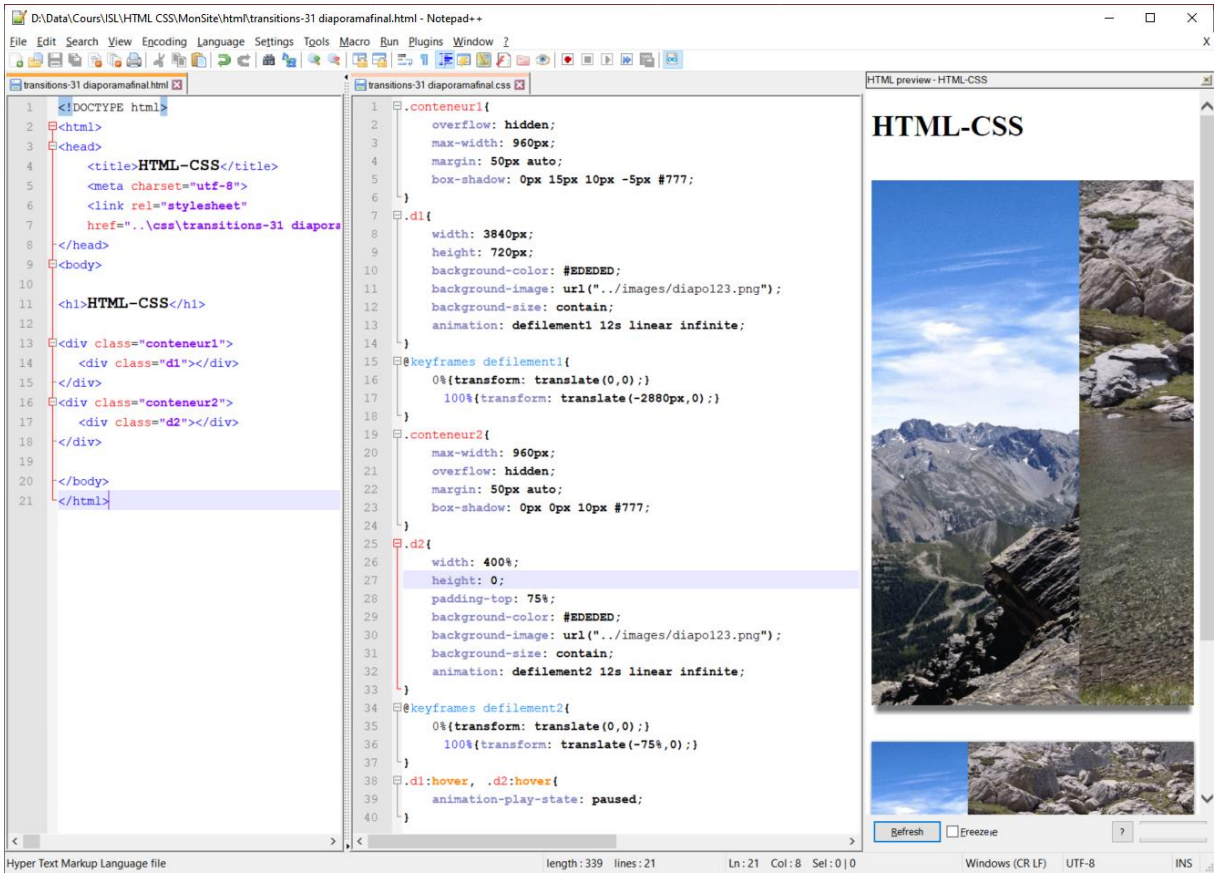
```css
@keyframes defilement2{
    0%{transform: translate(0,0);}
    100%{transform: translate(-75%,0);}
}
```

Ici, on indique un transform : translate(-75%,0) en fin d'animation. En effet, notre cadre fait 400% de largeur avec un ratio visible de 4 : 3. Notre image de fond va occuper 75% de cette largeur et va se répéter pour les derniers 25%.

Ainsi, transform : translate(-75%,0) nous ramène exactement à la position de l'image de fond en début d'animation et nous n'avons plus qu'à répéter l'animation à l'infini.

```css
.d2{
    width: 400%;
    height: 0;
    padding-top: 75%;
    background-color: #EDEDED;
    background-image: url("diapo123.png");
    background-size: contain;
    animation: defilement2 12s linear infinite;
}
```

Voici le code complet de ce diaporama :


D:\Data\Cours\ISL\HTML CSS\MonSite\html\transitions-31 diaporamafinal.html - Notepad++
transitions-31 diaporamafinal.html
transitions-31 diaporamafinal.css
HTML preview - HTML-CSS
HTML-CSS
Refresh
Freeze
Hyper Text Markup Language file
length : 339 lines : 21
Ln : 21 Col : 8 Sel : 0 | 0
Windows (CR LF)
UTF-8
INS