

HTML



CSS



Transitions, animations, transformations

Contents

Créer des transitions en CSS.....	5
Le fonctionnement d'une transition.....	5
La propriété CSS transition-property.....	7
La propriété CSS transition-duration.....	9
La propriété CSS transition-timing-function.....	9
La propriété CSS transition-delay.....	12
La propriété CSS transition.....	12
Transition et reverse transition.....	13
Créer des animations en CSS.....	16
Différence entre animations et transitions en CSS.....	16
La règle CSS @keyframes.....	16
La propriété animation-name.....	19
La propriété animation-duration.....	21
La propriété animation-timing-function.....	22
La propriété animation-iteration-count.....	23
La propriété animation-direction.....	24
La propriété animation-play-state.....	26
La propriété animation-delay.....	26
La propriété animation-fill-mode.....	27
La propriété animation.....	28
Créer des transformations en CSS.....	31
Définir une transformation en CSS.....	31
Exemples de transformations 2D.....	33
Modifier la taille ou l'échelle d'un élément avec scale().....	33
Déformer un élément avec skewX() et skewY().....	34
Effectuer une translation avec translate(X,Y).....	34
Effectuer une rotation avec rotate().....	35
Définir une matrice de transformation avec matrix().....	36
Appliquer plusieurs transformations d'un coup.....	36
Animer des transformations.....	37



INSTITUT SAINT-LAURENT

ENSEIGNEMENT DE PROMOTION SOCIALE

Baccalauréat en informatique

Les transformations 3D	38
------------------------------	----

Créer des transitions en CSS

Jusqu'à présent, à chaque fois qu'on définissait le comportement d'une propriété CSS pour un élément, la valeur définie s'appliquait directement.

Les transitions CSS vont nous permettre de modifier la valeur d'une propriété CSS de manière fluide et selon une durée que l'on va pouvoir définir. On va donc pouvoir définir deux valeurs pour une propriété (une première valeur de départ ou valeur par défaut et une seconde valeur d'arrivée) et faire en sorte que la valeur change progressivement de la valeur de départ à la valeur d'arrivée.

On va ainsi par exemple pouvoir changer progressivement la couleur des textes de nos éléments ou modifier la taille d'un élément, etc.

Pour créer des transitions en CSS, nous allons pouvoir utiliser les différentes propriétés de type **transition-*** ou la propriété raccourcie **transition**.

Dans cette nouvelle leçon, nous allons commencer par étudier les propriétés complètes qui sont les suivantes :

- **transition-property** ;
- **transition-duration** ;
- **transition-timing-function** ;
- **transition-delay**.

Nous terminerons avec la création d'une transition complète en utilisant la propriété raccourcie **transition**.

Le fonctionnement d'une transition

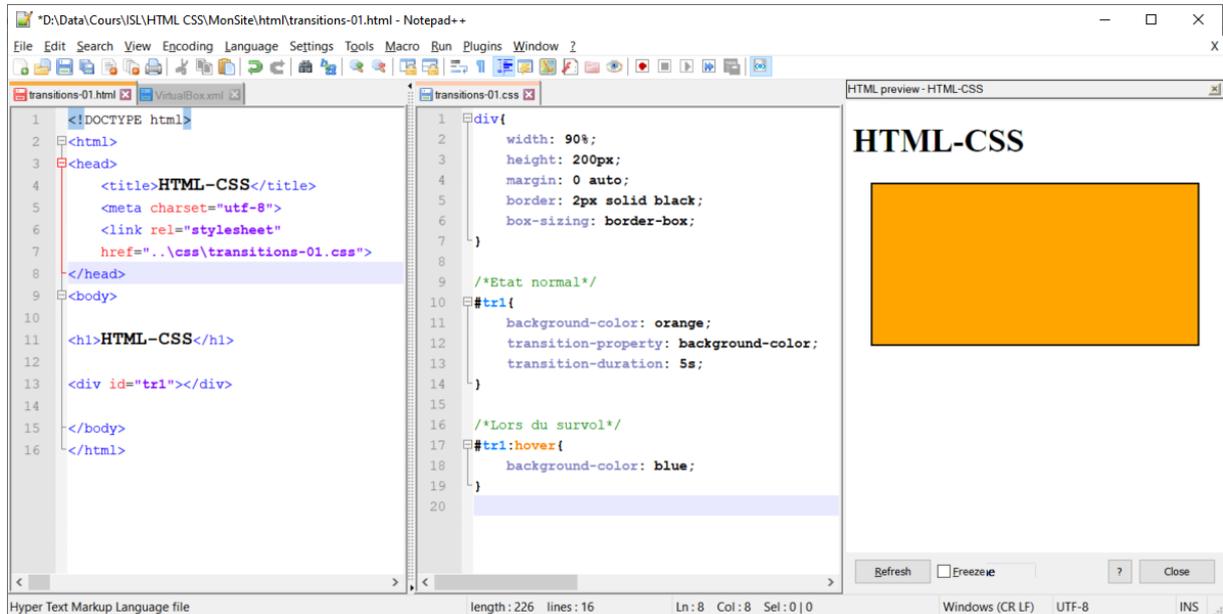
Le principe de base d'une transition en CSS est de modifier progressivement la valeur d'une propriété. Pour cela, nous allons devoir indiquer deux valeurs pour la propriété pour laquelle on souhaite créer une transition.

Le premier souci ici est qu'on ne va pas pouvoir passer deux valeurs pour une même propriété avec un même sélecteur CSS pour créer une transition, car en faisant cela seule la dernière valeur déclarée serait lue.

Nous avons ici deux solutions pour contourner cette contrainte. La première qui va nous permettre d'exploiter tout le potentiel des animations va être d'utiliser un langage dynamique comme le JavaScript qui peut mettre à jour le nom d'un attribut **class** par exemple en fonction de certains critères.

Comme nous n'avons pas étudié le JavaScript, nous allons mettre de côté cette première solution. La deuxième méthode va être d'utiliser les transitions avec les pseudo-classes, c'est-à-dire lors d'un changement d'état d'un élément HTML.

Nous allons par exemple pouvoir changer la couleur de fond d'un élément `div` lorsqu'un utilisateur passe sa souris dessus comme ci-dessous :



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-01.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div id="tr1"></div>
14
15 </body>
16 </html>

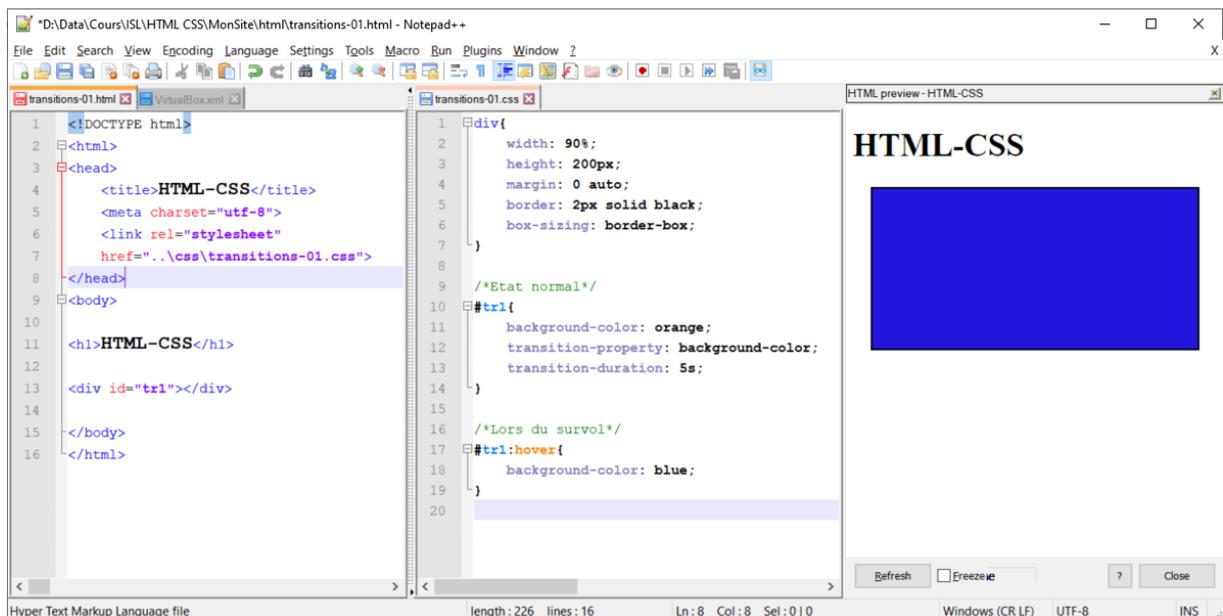
```

```

1 #div{
2 width: 90%;
3 height: 200px;
4 margin: 0 auto;
5 border: 2px solid black;
6 box-sizing: border-box;
7 }
8
9 /*Etat normal*/
10 #tr1{
11 background-color: orange;
12 transition-property: background-color;
13 transition-duration: 5s;
14 }
15
16 /*Lors du survol*/
17 #tr1:hover{
18 background-color: blue;
19 }
20

```

Au passage de la souris, la couleur de fond change progressivement vers le bleu en 5 secondes, et retour à l'orange lorsque la souris quitte le `div` :



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-01.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div id="tr1"></div>
14
15 </body>
16 </html>

```

```

1 #div{
2 width: 90%;
3 height: 200px;
4 margin: 0 auto;
5 border: 2px solid black;
6 box-sizing: border-box;
7 }
8
9 /*Etat normal*/
10 #tr1{
11 background-color: orange;
12 transition-property: background-color;
13 transition-duration: 5s;
14 }
15
16 /*Lors du survol*/
17 #tr1:hover{
18 background-color: blue;
19 }
20

```

Ici, on a utilisé deux propriétés pour créer une transition simple : la propriété `transition-property` qui sert à définir la ou les propriété(s) dont la valeur doit être modifiée progressivement et la propriété `transition-duration` qui indique le temps que va mettre la propriété à passer de sa valeur de départ à la valeur d'arrivée.

Comme vous pouvez le constater en observant le code CSS, on applique ces propriétés de transition à notre div dans son état normal et on définit deux couleurs de fond : orange dans l'état normal et bleu lorsque le div est survolé.

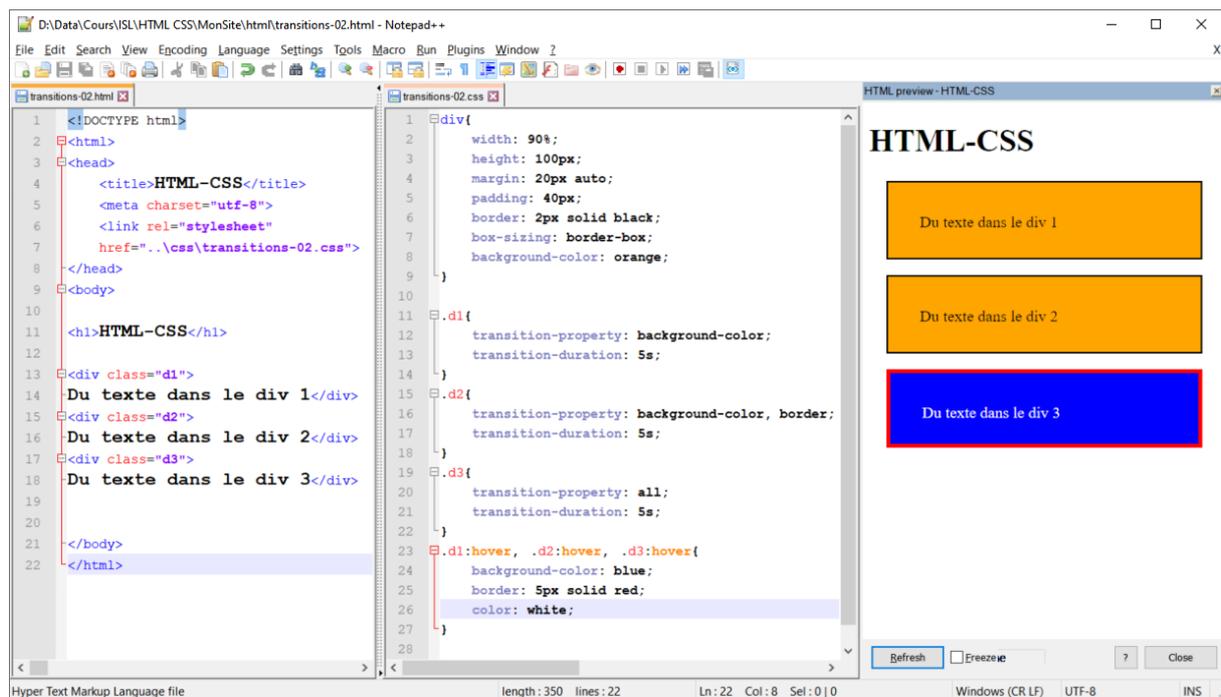
Lorsqu'on survole le **div**, sa couleur de fond va donc changer progressivement de l'orange vers le bleu sur une durée de 5 secondes. Notez que par défaut la transition se fait dans les deux sens : lorsqu'on sort du **div**, sa couleur de fond retourne progressivement vers l'orange.

Notez finalement qu'on ne va pas pouvoir créer des transitions avec toutes les propriétés car le concept de transitions est assez récent en CSS et donc le support n'est pas encore parfait. La plupart des propriétés vont tout de même pouvoir être animées.

La propriété CSS transition-property

La propriété **transition-property** va nous permettre de définir quelles propriétés vont être les cibles de nos transitions, c'est-à-dire quelles sont les propriétés dont la valeur va devoir changer progressivement.

On va pouvoir indiquer en valeur de **transition-property** soit le nom d'une propriété CSS pour laquelle on veut créer une transition, soit le nom de plusieurs propriétés CSS qui devront alors être séparées par des virgules, soit le mot clef **all** qui signifie que toutes les propriétés vont pouvoir être sujettes aux transitions (sous réserve qu'on indique ensuite une autre valeur d'arrivée pour créer la transition bien évidemment).



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-02.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1">
14 Du texte dans le div 1</div>
15 <div class="d2">
16 Du texte dans le div 2</div>
17 <div class="d3">
18 Du texte dans le div 3</div>
19 </body>
20 </html>
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2
```

Les transitions vont donc se faire lorsque l'utilisateur va passer sa souris sur un des div. Chacune des 3 transitions va durer 5 secondes.

Pour notre premier **div**, on applique la transition à la propriété **background-color** uniquement.

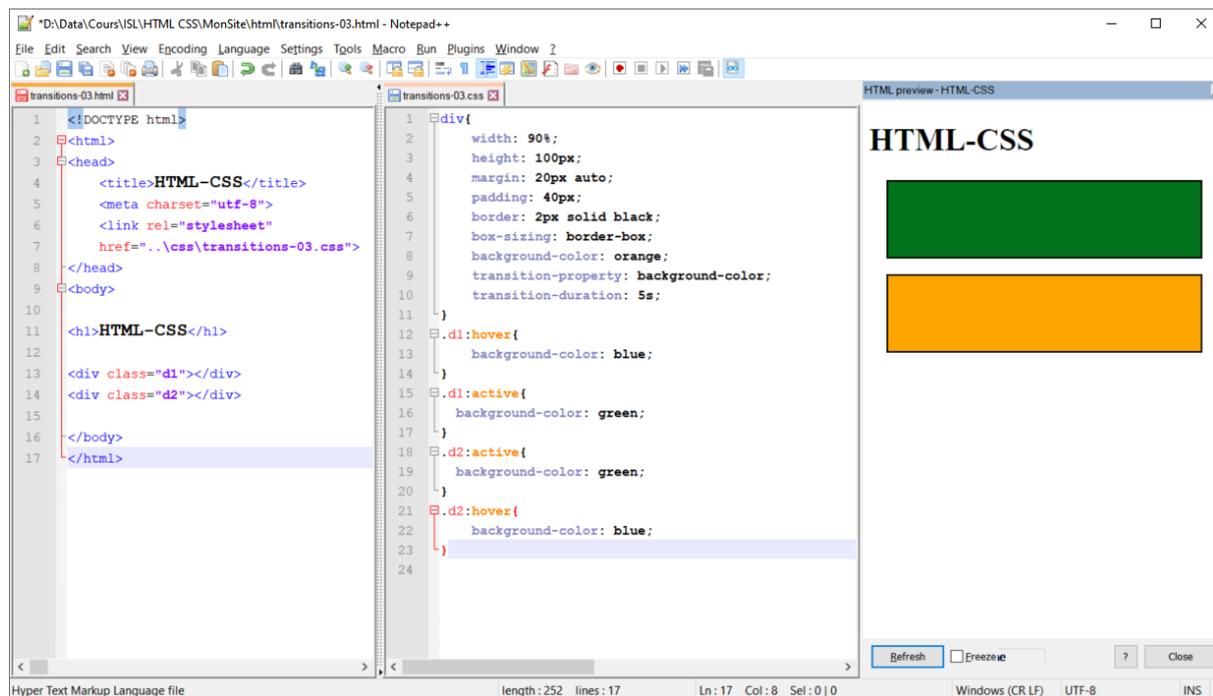
Pour notre deuxième **div**, on applique la transition aux propriétés **background-color** et **border**.

Pour notre troisième **div**, on applique une transition à toutes les propriétés qui ont plusieurs valeurs définies pour différents états de l'élément.

Notez que je n'ai pas défini de couleur de départ avec **color**. En effet, il n'est pas toujours strictement indispensable de définir des valeurs de départ pour les propriétés pour lesquelles on définit des transitions puisque la plupart des propriétés ont des valeurs par défaut. Pour **color**, la transition va donc se faire entre la valeur par défaut (noire) et la valeur d'arrivée spécifiée.

Ici, vous devez également bien comprendre qu'une transition ne va se déclencher que lors d'un changement d'état d'un élément. Ainsi, il est tout à fait possible de définir plus de deux valeurs pour une propriété à laquelle on souhaite appliquer une transition puisqu'on va pouvoir définir une valeur pour chaque état de l'élément.

Dans le cas où plusieurs changements d'état sont provoqués en même temps, la transition se fera vers la valeur de l'état déclaré en dernier en CSS. Par exemple, on peut définir une transition sur une couleur de fond en précisant une valeur d'arrivée pour l'état **hover** (survol) et l'état **active** (clicqué) :



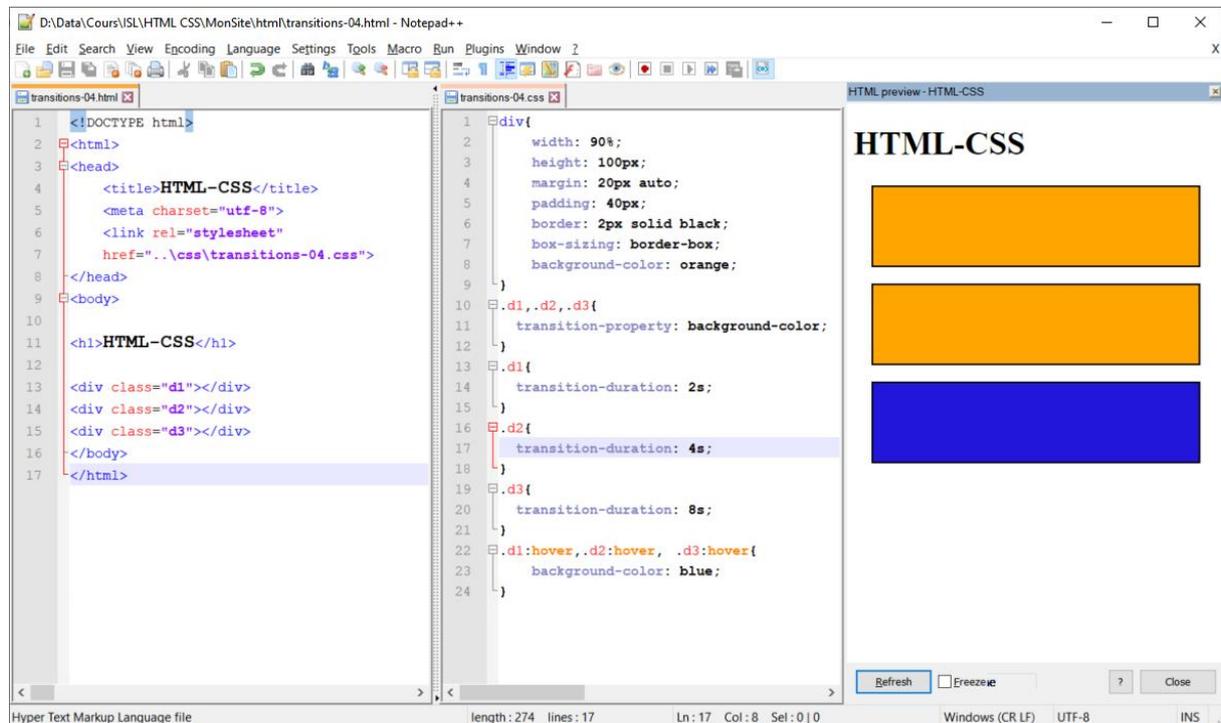
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-03.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12 <div class="d1"></div>
13 <div class="d2"></div>
14 </body>
15 </html>
```

```
1 div{
2 width: 90%;
3 height: 100px;
4 margin: 20px auto;
5 padding: 40px;
6 border: 2px solid black;
7 box-sizing: border-box;
8 background-color: orange;
9 transition-property: background-color;
10 transition-duration: 5s;
11 }
12 .d1:hover{
13 background-color: blue;
14 }
15 .d1:active{
16 background-color: green;
17 }
18 .d2:active{
19 background-color: green;
20 }
21 .d2:hover{
22 background-color: blue;
23 }
24 }
```

La propriété CSS transition-duration

La propriété transition-duration va nous permettre de définir le temps que vont mettre les propriétés passées à **transition-property** pour passer d'une valeur de départ à une valeur d'arrivée. Nous allons pouvoir lui passer un nombre de secondes en valeur.

La valeur par défaut de **transition-duration** est **0s** ce qui signifie que le changement de valeur des propriétés concernées par la transition sera immédiat.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-04.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1"></div>
14 <div class="d2"></div>
15 <div class="d3"></div>
16 </body>
17 </html>

1 div{
2 width: 90%;
3 height: 100px;
4 margin: 20px auto;
5 padding: 40px;
6 border: 2px solid black;
7 box-sizing: border-box;
8 background-color: orange;
9 }
10 .d1,.d2,.d3{
11 transition-property: background-color;
12 }
13 .d1{
14 transition-duration: 2s;
15 }
16 .d2{
17 transition-duration: 4s;
18 }
19 .d3{
20 transition-duration: 8s;
21 }
22 .d1:hover,.d2:hover,.d3:hover{
23 background-color: blue;
24 }
```

Ici, on applique une transition sur la couleur de fond de nos éléments HTML div lors du passage de la souris sur l'un d'eux. Chaque transition a une durée différente définie avec transition-duration ce qui signifie que la couleur de fond mettra plus ou moins de temps à passer de sa couleur de départ à sa couleur d'arrivée selon le div.

La propriété CSS transition-timing-function

La propriété **transition-timing-function** va nous permettre de choisir la vitesse de la transition au sein de celle-ci. Nous allons ainsi pouvoir créer des transitions totalement linéaires ou, au contraire, créer des transitions qui vont s'accélérer ou ralentir au milieu. Nous allons pouvoir passer les valeurs suivantes à cette propriété :

- **ease** : valeur par défaut. Permet de créer une transition relativement lente au début puis qui s'accélère au milieu et qui se termine lentement ;
- **linear** : permet de créer une transition totalement linéaire c'est-à-dire qui va aller à la même vitesse du début à la fin ;
- **ease-in** : permet de créer une transition avec un départ lent puis qui s'accélère ensuite ;

- **ease-out** : permet de créer une transition qui va ralentir à la fin ;
- **ease-in-out** : permet de créer une transition lente au début puis qui s'accélère au milieu et qui se termine lentement. Ressemble fortement à **transition-timing-function : ease** mais démarre plus lentement ;
- **cubic-bezier(x1,y1,x2,y2)** : sert à créer une transition à la vitesse totalement personnalisée en renseignant une courbe de Bézier.

Notre but n'est ici bien évidemment pas de faire un cours sur les courbes de Bézier. Vous pouvez simplement retenir les équivalents mot-clef/Bézier suivants qui peuvent se révéler utiles :

- **transition-timing-function : ease** est équivalent à **transition-timing-function : cubic-bezier(0.25, 0.1, 0.25, 1)** ;
- **transition-timing-function : ease-in** est équivalent à **transition-timing-function : cubic-bezier(0.42, 0, 1, 1)** ;
- **transition-timing-function : ease-out** est équivalent à **transition-timing-function : cubic-bezier(0, 0, 0.58, 1)** ;
- **transition-timing-function : ease-in-out** est équivalent à **transition-timing-function : cubic-bezier(0.42, 0, 0.58, 1)**.

```

D:\Data\Cours\ISL\HTML CSS\MonSite\html\transitions-05.html - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
transitions-05.html transitions-05.css
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-05.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1">
14 transition-timing-function:
15 ease</div>
16 <div class="d2">
17 transition-timing-function:
18 ease-in</div>
19 <div class="d3">
20 transition-timing-function:
21 ease-out</div>
22 <div class="d4">
23 transition-timing-function:
24 ease-in-out</div>
25 <div class="d5">
26 transition-timing-function:
27 linear</div>
28 <div class="d6">
29 transition-timing-function:
30 cubic-bezier(0.8, 0, 1, 1)
31 </div>
32
33 </body>
34 </html>
1 div{
2 width: 50%;
3 height: 60px;
4 margin: 20px auto;
5 border: 2px solid black;
6 box-sizing: border-box;
7 text-align: center;
8 padding: 15px;
9 }
10
11 .d1, .d2, .d3, .d4, .d5, .d6{
12 background-color: orange;
13 transition-property: width;
14 transition-duration: 2s;
15 }
16
17 .d1{
18 transition-timing-function: ease;
19 }
20
21 .d2{
22 transition-timing-function: ease-in;
23 }
24
25 .d3{
26 transition-timing-function: ease-out;
27 }
28
29 .d4{
30 transition-timing-function: ease-in-out;
31 }
32
33 .d5{
34 transition-timing-function: linear;
35 }
36
37 .d6{
38 transition-timing-function: cubic-bezier(0.6, 0, 1, 1);
39 }
40
41 .d1:hover, .d2:hover, .d3:hover, .d4:hover, .d5:hover, .d6:hover{
42 width: 100%;
43 }
44
45
Hyper Text Markup Language file length: 603 lines: 34 Ln: 17 Col: 24 Sel: 0 | 0 Windows (CR LF) UTF-8 INS

```

HTML-CSS

Fichier | D:/Data/Cours/ISL/HTML%20CSS/MonSite/html/transitions-05.html

HTML-CSS

transition-timing-function: ease

transition-timing-function: ease-in

transition-timing-function: ease-out

transition-timing-function: ease-in-out

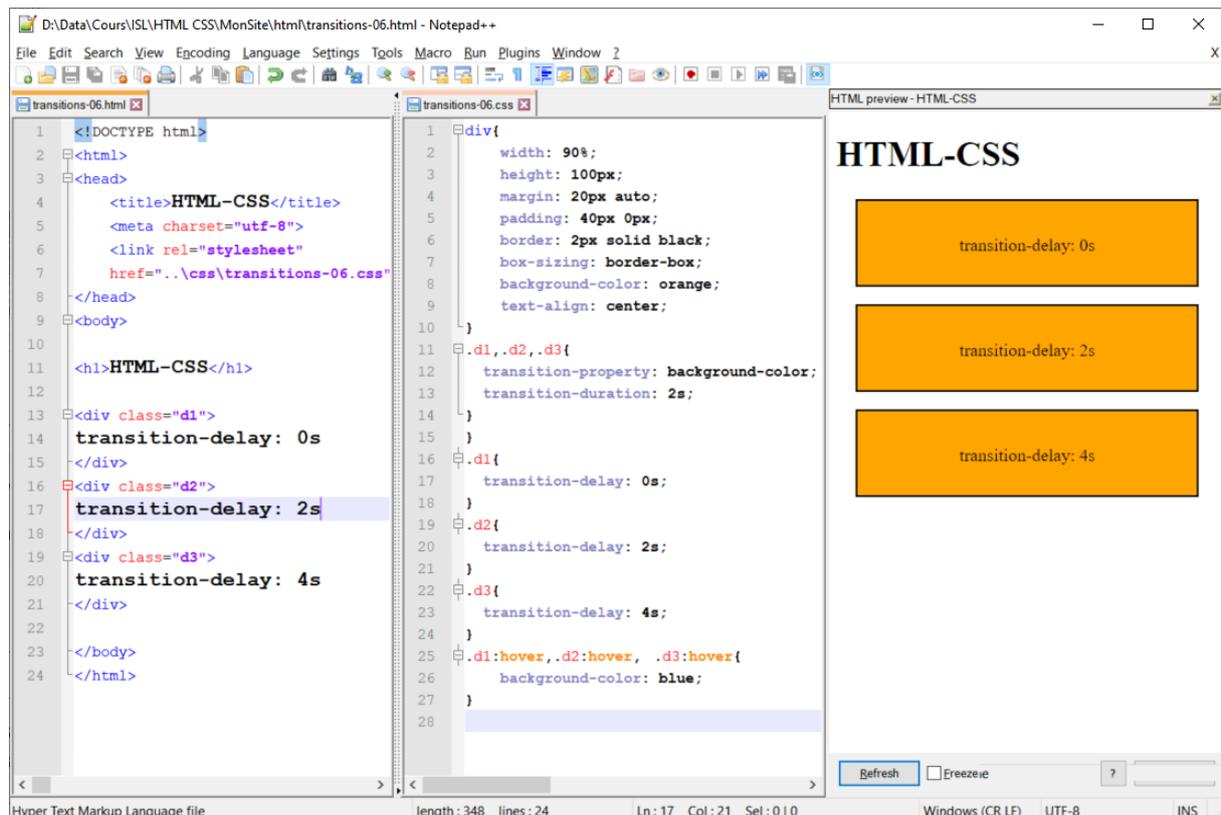
transition-timing-function: linear

transition-timing-function: cubic-bezier(0.8, 0, 1, 1)

La propriété CSS transition-delay

La propriété **transition-delay** va nous permettre de définir quand la transition doit commencer à partir du moment où la nouvelle valeur est passée aux propriétés concernées par la transition. On va pouvoir lui passer une valeur en secondes.

La valeur par défaut est 0s (la transition se lance dès qu'une nouvelle valeur est définie pour une propriété à laquelle on a appliqué la transition).



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-06.css"
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1">
14 transition-delay: 0s
15 </div>
16 <div class="d2">
17 transition-delay: 2s
18 </div>
19 <div class="d3">
20 transition-delay: 4s
21 </div>
22
23 </body>
24 </html>

```

```

1 div{
2 width: 90%;
3 height: 100px;
4 margin: 20px auto;
5 padding: 40px 0px;
6 border: 2px solid black;
7 box-sizing: border-box;
8 background-color: orange;
9 text-align: center;
10 }
11 .d1,.d2,.d3{
12 transition-property: background-color;
13 transition-duration: 2s;
14 }
15
16 .d1{
17 transition-delay: 0s;
18 }
19 .d2{
20 transition-delay: 2s;
21 }
22 .d3{
23 transition-delay: 4s;
24 }
25 .d1:hover,.d2:hover,.d3:hover{
26 background-color: blue;
27 }
28

```

La propriété CSS transition

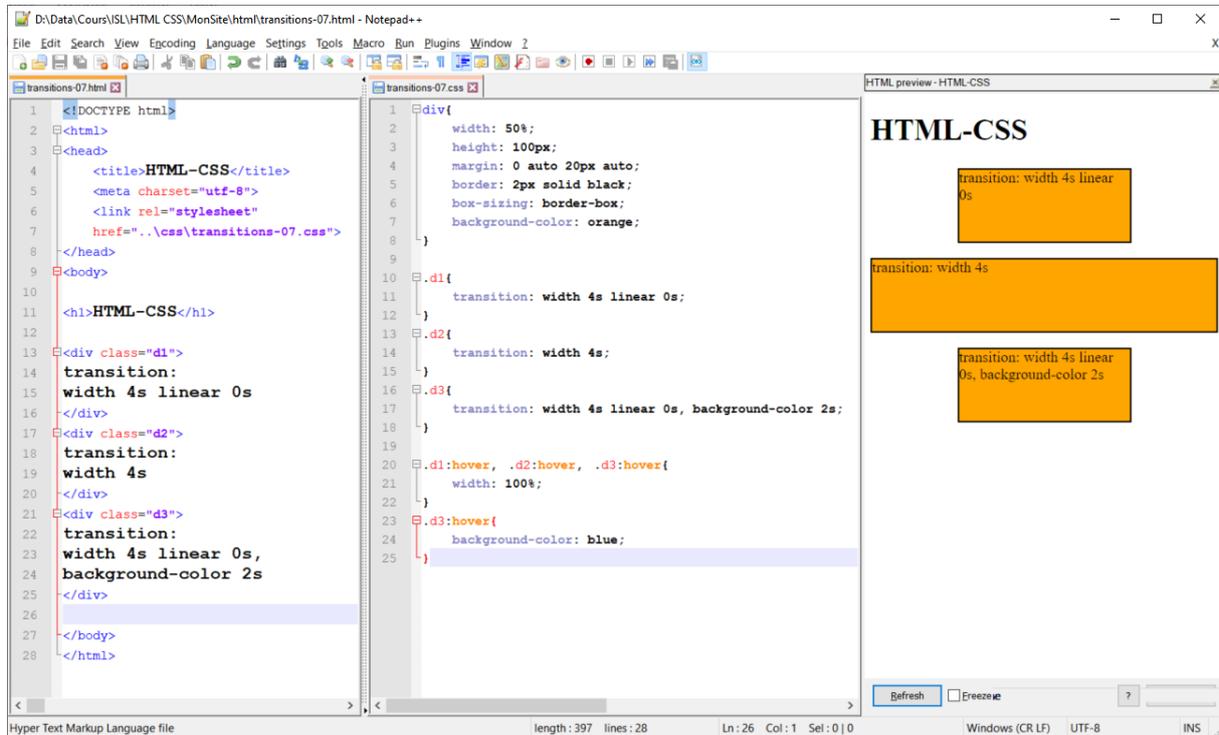
La propriété CSS **transition** est la notation raccourcie des quatre propriétés étudiées précédemment. On va pouvoir lui passer les différentes valeurs des propriétés précédentes à la suite pour créer une transition complète.

La première durée renseignée dans **transition** définira la durée de la transition. Je vous conseille de suivre l'ordre de déclaration des valeurs suivant pour être sûr que **transition** fonctionne bien :

1. La valeur relative à **transition-property** ;
2. La valeur relative à **transition-duration** ;
3. La valeur relative à **transition-timing-function** ;
4. La valeur relative à **transition-delay**.

A noter que seules les valeurs relatives aux propriétés **transition-property** et **transition-duration** doivent être obligatoirement renseignées pour créer une transition visible. Les valeurs relatives aux propriétés **transition-timing-function** et **transition-delay** sont facultatives et si rien n'est renseigné les valeurs par défaut seront utilisées.

Notez également que pour créer plusieurs transitions pour plusieurs propriétés différentes avec **transition** il va suffire de séparer les différentes déclarations par une virgule.



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-07.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1">
14 transition:
15 width 4s linear 0s
16 </div>
17 <div class="d2">
18 transition:
19 width 4s
20 </div>
21 <div class="d3">
22 transition:
23 width 4s linear 0s,
24 background-color 2s
25 </div>
26
27 </body>
28 </html>

```

```

1 div{
2 width: 50%;
3 height: 100px;
4 margin: 0 auto 20px auto;
5 border: 2px solid black;
6 box-sizing: border-box;
7 background-color: orange;
8 }
9
10 .d1{
11 transition: width 4s linear 0s;
12 }
13 .d2{
14 transition: width 4s;
15 }
16 .d3{
17 transition: width 4s linear 0s, background-color 2s;
18 }
19
20 .d1:hover, .d2:hover, .d3:hover{
21 width: 100%;
22 }
23
24 .d3:hover{
25 background-color: blue;
26 }

```

The screenshot shows a Notepad++ window with three panes. The left pane contains the HTML code, the middle pane contains the CSS code, and the right pane shows a live preview of the web page. The preview shows three orange boxes with different transition effects: the first box transitions from 50% to 100% width over 4 seconds with a linear timing function and a 0s delay; the second box transitions from 50% to 100% width over 4 seconds with the default ease timing function and 0s delay; the third box transitions from 50% to 100% width over 4 seconds with a linear timing function and 0s delay, and also transitions from orange to blue background color over 2 seconds.

Ici, on renseigne toutes les valeurs pour notre première transition dans la propriété **transition**. Pour notre deuxième transition, en revanche, on omet les deux dernières valeurs. Ce seront donc les valeurs par défaut **ease** et **0s** qui seront utilisées.

Finalement, nous créons deux transitions dans notre dernier **div**. Pour faire cela, on se contente de séparer les différentes déclarations pour chacune de nos transitions par une virgule dans notre propriété **transition**.

Transition et reverse transition

Par défaut, lorsqu'on crée une transition entre les valeurs d'une ou de plusieurs propriétés, la transition va se faire dans les deux sens :

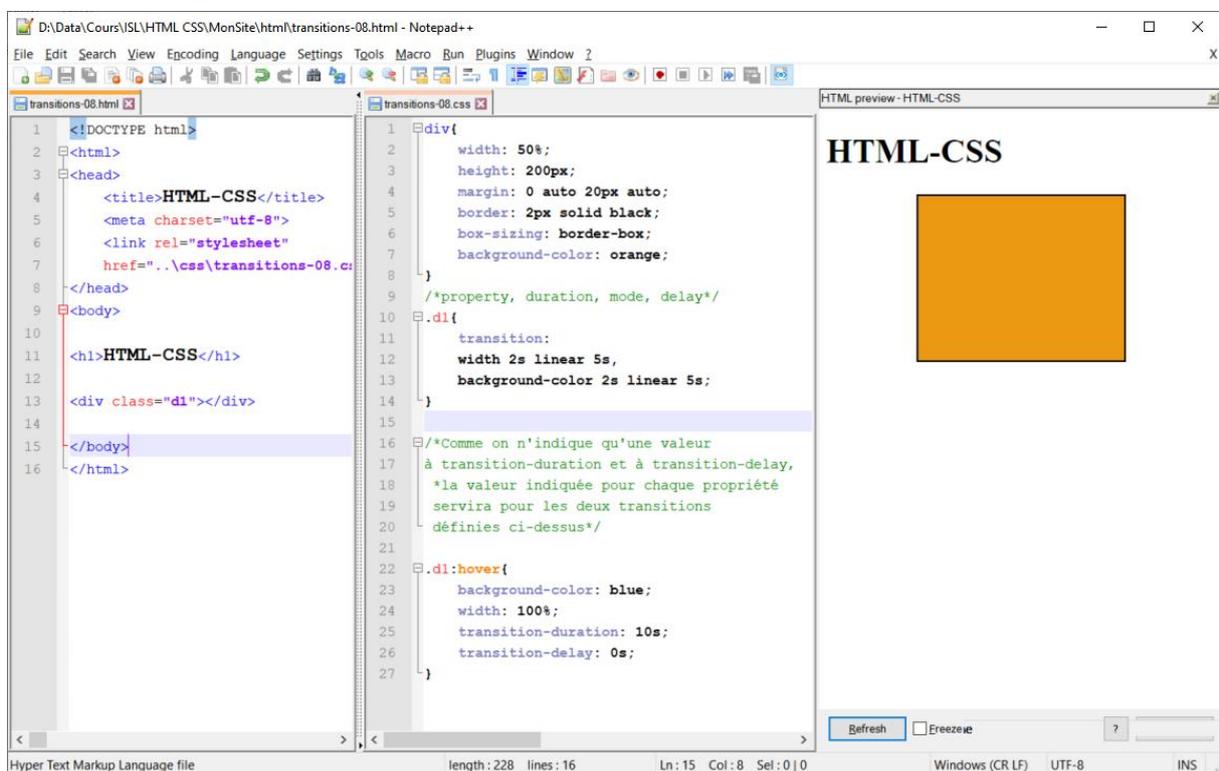
1. A partir de la valeur de départ vers la valeur d'arrivée lors du changement d'état d'un élément ;

2. A partir de la valeur actuelle vers la valeur de départ lorsque l'élément retourne dans son état initial.

Par exemple, si on crée une transition qui doit se déclencher lors du passage de la souris sur un élément, dès que l'on va déplacer notre souris en dehors de l'élément les propriétés concernées par la transition vont reprendre petit à petit leurs valeurs initiales en effectuant une transition inversée vers leurs valeurs initiales plutôt que d'être redéfinie brutalement sur leurs valeurs de départ.

Nous allons pouvoir également pouvoir gérer la transition de retour et notamment la durée de cette transition et son délai. Pour faire cela, nous allons devoir préciser en plus de nos valeurs de transition classiques des valeurs de transition dans l'état qui va déclencher la transition (à l'intérieur d'un `:hover` par exemple pour une transition qui doit démarre lorsque l'utilisateur passe sa souris sur un élément).

Attention ici : les valeurs précisées dans cet état seront utilisées pour la transition de départ et ce sont les valeurs précisées au départ qui seront utilisées pour la transition de retour. Regardez l'exemple suivant pour bien comprendre :



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-08.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1"></div>
14
15 </body>
16 </html>

```

```

1 div{
2 width: 50%;
3 height: 200px;
4 margin: 0 auto 20px auto;
5 border: 2px solid black;
6 box-sizing: border-box;
7 background-color: orange;
8 }
9 /*property, duration, mode, delay*/
10 .d1{
11 transition:
12 width 2s linear 5s,
13 background-color 2s linear 5s;
14 }
15
16 /*Comme on n'indique qu'une valeur
17 à transition-duration et à transition-delay,
18 *la valeur indiquée pour chaque propriété
19 servira pour les deux transitions
20 définies ci-dessus*/
21
22 .d1:hover{
23 background-color: blue;
24 width: 100%;
25 transition-duration: 10s;
26 transition-delay: 0s;
27 }

```

Ici, on crée une transition sur la couleur de fond et la taille de notre `div class="d1"` qui doit démarrer lorsqu'un utilisateur passe sa souris sur notre `div`.

Cette première transition ici va durer 10 secondes et avoir un délai de 0s. Dès que l'utilisateur sort sa souris de l'élément, les valeurs de nos propriétés vont revenir à leur état normal en effectuant une transition de retour. La transition de retour va durer 2 secondes et va démarrer après un délai de 5 secondes.



INSTITUT SAINT-LAURENT

ENSEIGNEMENT DE PROMOTION SOCIALE

Baccalauréat en informatique

La relation entre les valeurs et les transitions peut paraître contre intuitif à priori, donc faites bien attention à bien retenir cela !

Créer des animations en CSS

Les animations vont, comme les transitions, nous permettre de modifier la valeur d'une propriété progressivement mais en utilisant cette fois-ci des **keyframes**.

Nous allons pouvoir gérer le comportement des animations en CSS en définissant la durée d'animation, le nombre de répétition et le comportement de répétition.

De manière similaire aux animations, toutes les propriétés ne vont pas pouvoir être animées. Cependant, la grande majorité des propriétés courantes vont pouvoir l'être.

Pour définir une animation en CSS nous allons pouvoir utiliser les propriétés de type **animation-*** ou la notation raccourcie **animation** avec une règle **@keyframes** qui va contenir les propriétés à animer et leurs valeurs.

Différence entre animations et transitions en CSS

Les transitions et les animations permettent toutes les deux de modifier la valeur de propriétés de manière progressive, au cours du temps. Cependant, la façon dont vont procéder les transitions et les animations pour arriver à cela ne va pas être la même.

La grande différence entre les transitions et les animations en CSS est que les animations nous laissent à la fois une plus grande liberté et un plus grand contrôle sur le déclenchement et la progression du changement de valeur des propriétés animées.

En effet, dans le cas d'une transition, nous ne pouvons que préciser une valeur de départ et une valeur d'arrivée pour les propriétés pour lesquelles nous souhaitons créer notre transition et n'avons pas véritablement de contrôle précis sur la transition en soi tandis que dans le cas d'une animation nous allons pouvoir indiquer de manière explicite comment la « transition » entre les différentes valeurs doit se passer et pouvoir préciser différentes valeurs intermédiaires.

En cela, les animations offrent davantage de contrôle sur le changement de valeurs des propriétés concernées par l'animation puisqu'on va pouvoir contrôler ce changement de valeur dans son ensemble. Elles vont donc notre choix de prédilection lorsqu'on voudra créer des effets plus complexes ou précis.

De plus, nous n'allons plus devoir attendre un changement d'état d'un élément pour modifier la valeur d'une de ses propriétés avec les animations. En effet, nous allons pouvoir lancer une animation dès le chargement de la page ou selon un autre évènement.

La règle CSS @keyframes

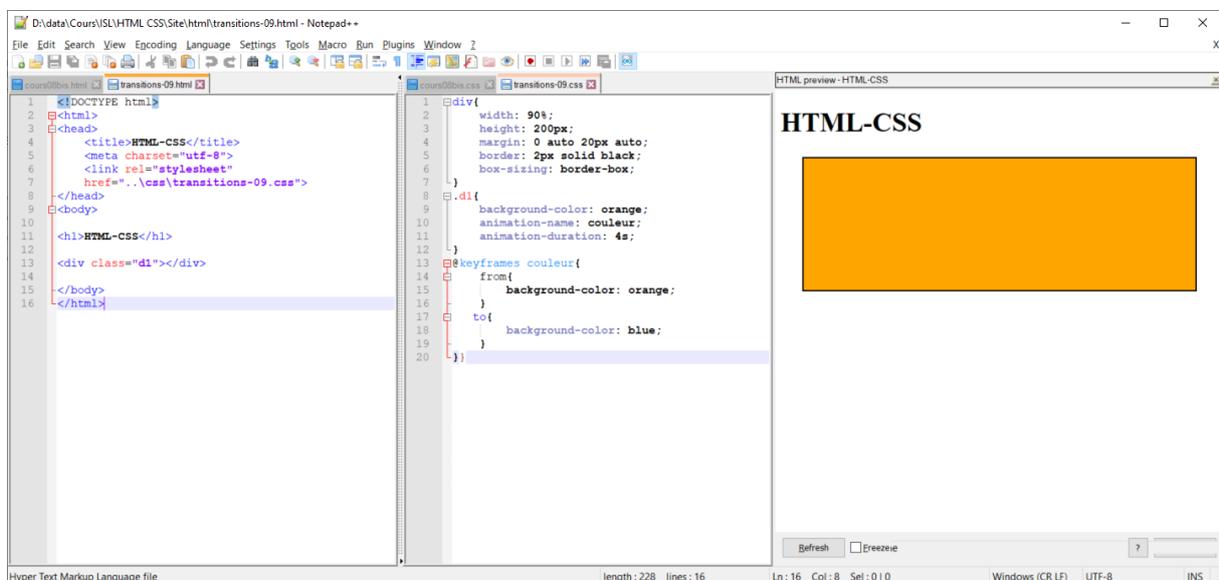
La règle **@keyframes** va nous permettre d'indiquer quelles propriétés doivent être animées et comment elles doivent l'être. Nous allons pouvoir dans notre règle **@keyframes** préciser différentes valeurs auxquelles les propriétés animées doivent parvenir à certains stades de l'animation.

La règle **@keyframes** va donc nous permettre de définir différents stades ou étapes pour notre animation et c'est ce qui va nous permettre d'avoir un contrôle total sur la progression de l'animation.

Nous allons toujours devoir donner un nom à une règle **@keyframes** qu'on va ensuite réutiliser dans notre animation pour lui indiquer les propriétés à animer et comment les animer. Dans notre règle **@keyframes**, nous allons tout simplement renseigner les propriétés à animer et les différentes valeurs qu'elles doivent atteindre à certains moments de l'animation.

Je vous rappelle ici que pour créer une animation fonctionnelle en CSS, nous allons d'une part devoir définir notre règle **@keyframes** et d'autre part utiliser les propriétés de type **animation-*** ou la notation raccourcie **animation** pour définir les caractéristiques propres à notre animation (nom, durée, etc.).

Regardez plutôt l'exemple suivant pour bien comprendre (l'animation va se lancer immédiatement, n'hésitez pas à la relancer en cliquant sur F5 pour rafraîchir la page) :



The screenshot shows a Notepad++ window with three panes. The left pane shows the HTML code for a page titled 'HTML-CSS', including a link to a CSS file. The middle pane shows the CSS code, which includes a `div` style with dimensions and a `@keyframes couleur` rule that animates the `background-color` from orange to blue over 4 seconds. The right pane is a live preview titled 'HTML-CSS' showing a yellow rectangle on a white background.

Ici, vous pouvez déjà remarquer une différence intéressante entre les animations et les transitions : une fois l'animation terminée, les propriétés animées retrouvent immédiatement leur valeur « normale ».

Regardons maintenant en détail notre règle **@keyframes**. Ici, on lui donne le nom `couleur` et on précise dedans les propriétés qui vont être animées (en l'occurrence la propriété **background-color**).

Plus précisément, on indique ici la valeur de départ pour notre ou nos propriété(s) que l'on souhaite animer à l'intérieur d'un bloc **from{}** qui signifie « à partir de » et la valeur d'arrivée pour notre ou nos propriété(s) que l'on souhaite animer à l'intérieur d'un bloc **to{}** qui signifie « vers ».

A ce niveau, il y a une chose que vous devez bien comprendre : la valeur que l'on précise dans le bloc **from{}** est bien la valeur de départ de l'animation, c'est-à-dire la valeur avec laquelle la propriété doit

démarrer son animation et non pas sa valeur « normale » c'est-à-dire sa valeur en dehors du temps de l'animation (avant le lancement de l'animation et après sa fin).

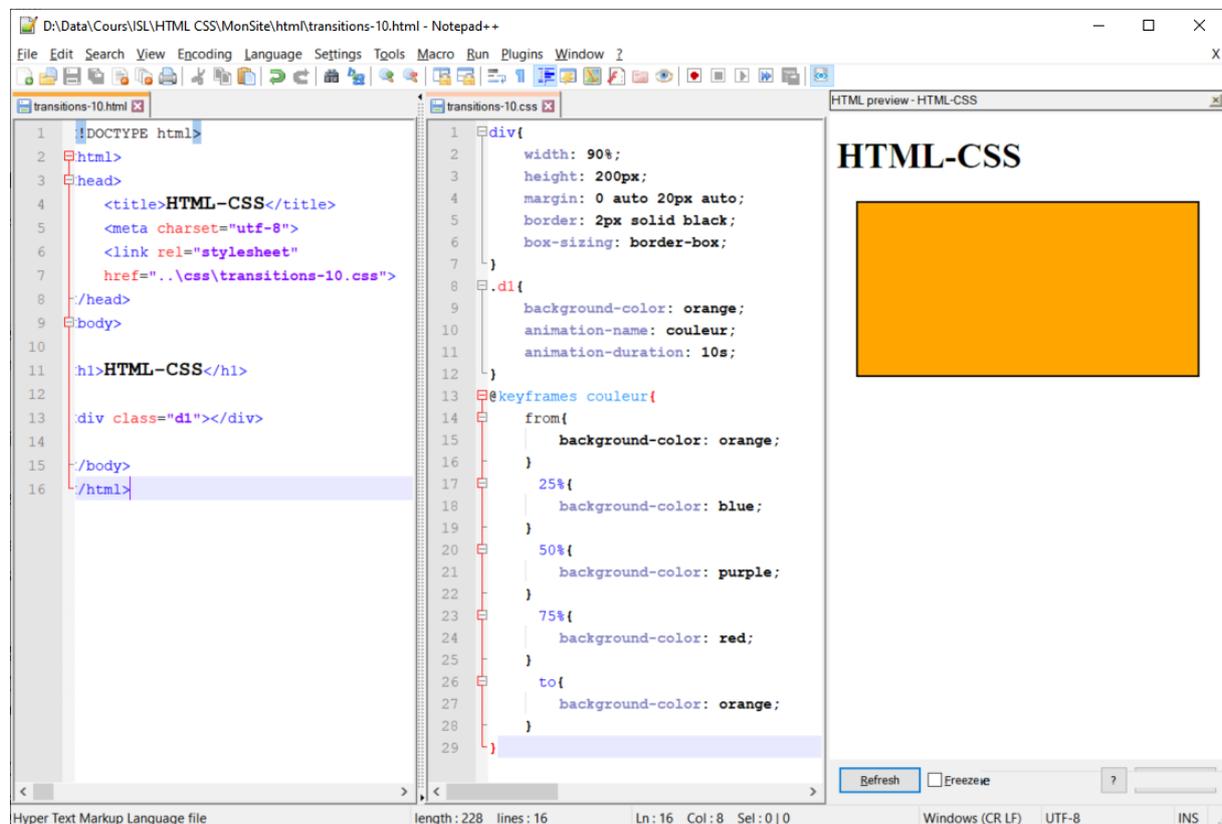
C'est pour cela que j'ai également précisé une couleur de fond en dehors de ma règle `@keyframes` afin de préciser la valeur que doit avoir ma propriété lorsqu'elle n'est pas animée.

Une fois notre règle `@keyframes` définie, nous allons devoir la lier ou l'attribuer à une animation. Pour cela, nous allons également devoir déclarer une animation et définir son comportement.

C'est ce que j'ai fait ici en déclarant deux propriétés animation-name à laquelle j'ai passé notre règle `@keyframes` et animation-duration qui me permet de définir la durée de l'animation. Par défaut, l'animation ne va s'exécuter qu'une fois.

Dans l'exemple précédent, utiliser une animation n'a aucun intérêt par rapport à une simple transition car cette animation est très simple. Cependant, rappelez-vous qu'un des grands avantages des animations par rapport aux transitions est qu'on va pouvoir dans notre règle `@keyframes` préciser différentes valeurs que nos propriétés devront atteindre à un moment choisi de l'animation.

On va ainsi entre autres très facilement pouvoir créer une boucle sur les valeurs d'une propriété pendant une animation comme dans l'exemple suivant :



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-10.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1"></div>
14
15 </body>
16 </html>

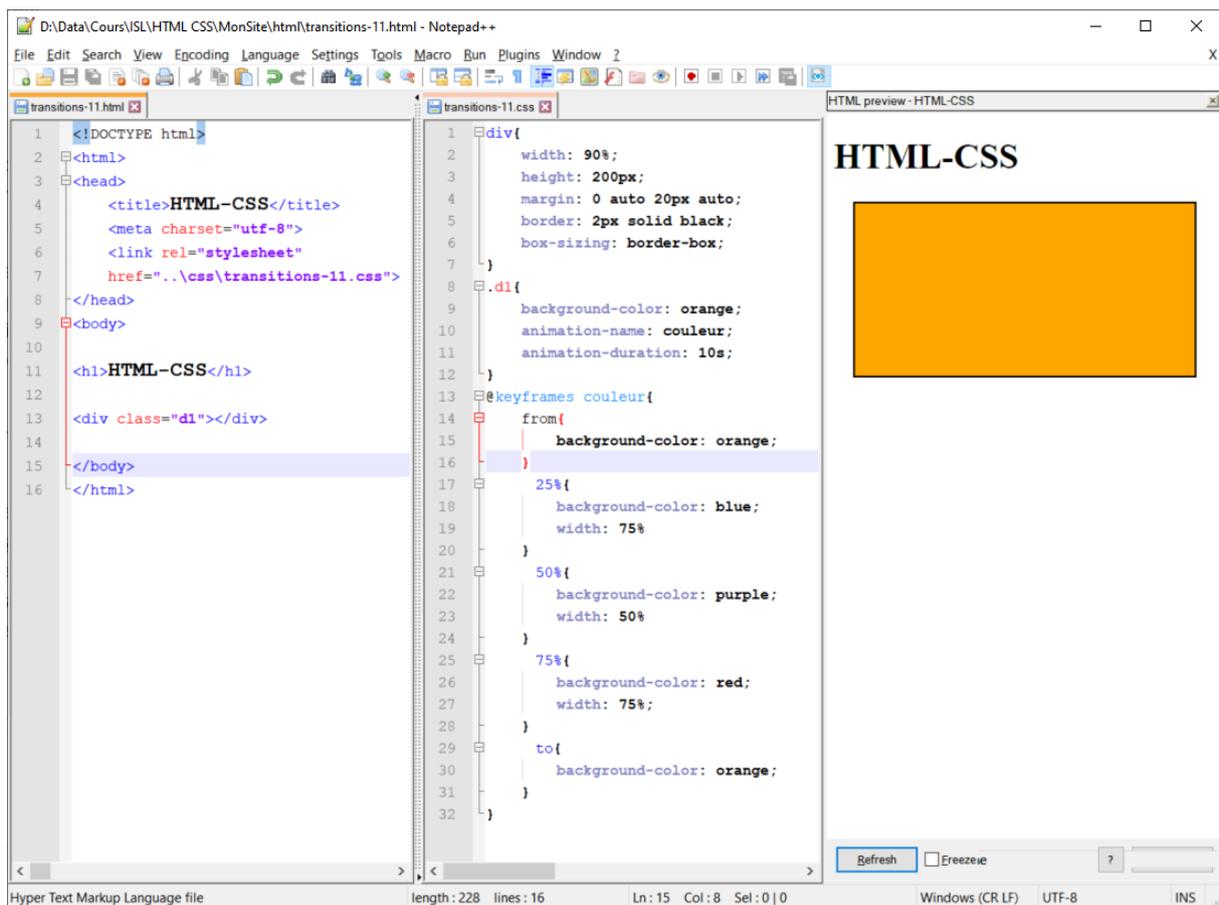
1 div{
2 width: 90%;
3 height: 200px;
4 margin: 0 auto 20px auto;
5 border: 2px solid black;
6 box-sizing: border-box;
7 }
8
9 .d1{
10 background-color: orange;
11 animation-name: couleur;
12 animation-duration: 10s;
13 }
14 @keyframes couleur{
15 from{
16 background-color: orange;
17 }
18 25%{
19 background-color: blue;
20 }
21 50%{
22 background-color: purple;
23 }
24 75%{
25 background-color: red;
26 }
27 to{
28 background-color: orange;
29 }
```

Ici, en plus de mes mots clefs `from` et `to`, j'ai indiqué des pourcentages dans ma règle `@keyframes`. Ces pourcentages correspondent à l'état d'avancement de l'animation. 50% marque donc le milieu de l'animation, c'est-à-dire au bout de 5 secondes pour une animation qui doit durer 10 secondes ;

25% correspond à 25% du temps total de l'animation, soit 2,5 secondes pour une animation de 10 secondes et etc. Ensuite, on indique la valeur que doit avoir notre propriété animée pour chaque niveau d'avancement de l'animation défini.

Le grand avantage des animations encore une fois est que nous allons pouvoir définir autant d'étapes que l'on souhaite pour contrôler plus ou moins le l'avancement de notre animation plus. J'attribue ici la même valeur à ma propriété **background-color** au début et en fin d'animation pour donner l'impression que la couleur boucle sur elle-même.

Nous allons de plus pouvoir préciser plusieurs propriétés à animer d'un coup et leur attribuer différentes valeurs à différents moments de l'animation au sein d'une même règle **@keyframes** :



The screenshot shows a Notepad++ window with three panes. The left pane shows the HTML code for 'transitions-11.html', which includes a title 'HTML-CSS' and a div with class 'd1'. The middle pane shows the CSS code for 'transitions-11.css', which defines a keyframe animation named 'couleur' with four steps: 0% (orange), 25% (blue), 50% (purple), 75% (red), and 100% (orange). The right pane shows a live preview of the HTML-CSS page, displaying a yellow square that animates through the colors defined in the CSS.

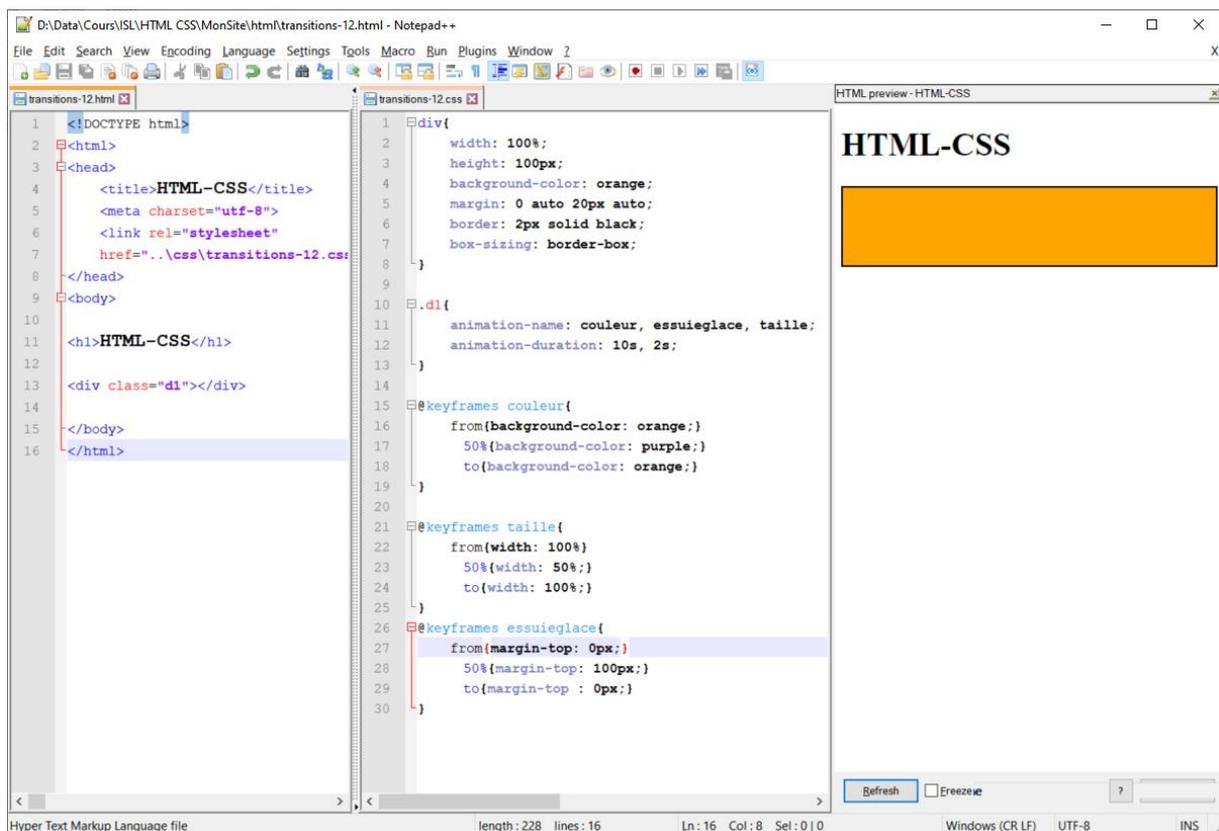
Maintenant que nous avons vu et compris le rôle de la règle **@keyframes** dans une animation, il est temps d'étudier les différentes propriétés animation-* qui vont nous permettre de gérer notre animation en soi avant de terminer avec la propriété raccourcie animation.

La propriété animation-name

La propriété **animation-name** va nous permettre de définir une liste d'animations qui doivent s'exécuter. On va donc lui passer un ou plusieurs noms qui devront correspondre aux noms des règles **@keyframes** qui définissent les propriétés à animer et les différentes valeurs qu'elle doit avoir pendant l'animation.

Si on fournit plusieurs noms à la propriété **animation-name**, alors il faudra définir à minima la durée de l'animation pour chaque animation avec la propriété **animation-duration** si on veut des animations fonctionnelles. Dans le cas où l'on fournit moins de valeurs à **animation-duration** qu'à **animation-name**, alors les animations supplémentaires vont réutiliser les valeurs de **animation-duration** dans l'ordre.

Par exemple, si on définit 5 animations avec **animation-name** et qu'on ne donne que 3 valeurs à **animation-duration**, alors la quatrième animation reprendra la première valeur de **animation-duration** et la cinquième animation reprendra sa deuxième valeur.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-12.css">
8 </head>
9 <body>
10 <h1>HTML-CSS</h1>
11 <div class="d1"></div>
12
13 </body>
14 </html>
```

```
1 div{
2 width: 100%;
3 height: 100px;
4 background-color: orange;
5 margin: 0 auto 20px auto;
6 border: 2px solid black;
7 box-sizing: border-box;
8 }
9
10 .d1{
11 animation-name: couleur, essuieglace, taille;
12 animation-duration: 10s, 2s;
13 }
14
15 @keyframes couleur{
16 from{background-color: orange;}
17 50%{background-color: purple;}
18 to{background-color: orange;}
19 }
20
21 @keyframes taille{
22 from{width: 100%}
23 50%{width: 50%;}
24 to{width: 100%;}
25 }
26
27 @keyframes essuieglace{
28 from{margin-top: 0px;}
29 50%{margin-top: 100px;}
30 to{margin-top: 0px;}
31 }
```

Dans l'exemple précédent, on définit 3 règles **@keyframes** qui définissent chacune des changements de valeurs pour une propriété. Dans le cas présent, on aurait aussi bien pu tout mettre dans une seule règle **@keyframes**. Cependant, il est généralement considéré comme une bonne pratique d'avoir une règle **@keyframes** pour une propriété puisque cela va nous permettre de nous resservir indépendamment d'une règle ou d'une autre pour l'appliquer à un élément ou à un autre.

Je passe ensuite les trois noms des animations que je souhaite exécuter à ma propriété **animation-name**. Ici, vous pouvez remarquer que je n'indique que deux valeurs de durée d'animation dans ma propriété **animation-duration**.

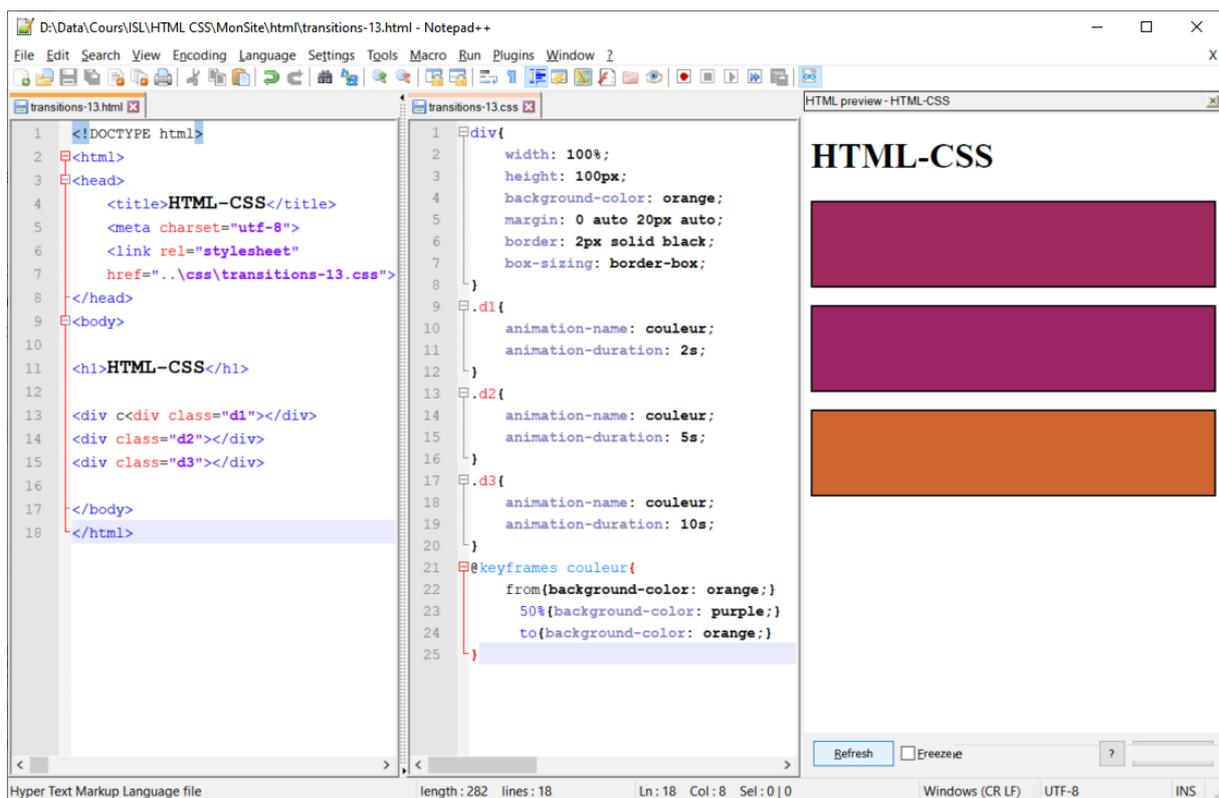
La première animation déclarée dans **animation-name** se servira de la première valeur, la deuxième de la deuxième valeur et la troisième à nouveau de la première valeur. L'animation « couleur » va

donc durer 10 secondes, l'animation « essuieglace » va durer 2 secondes et l'animation « taille » va durer 10 secondes.

Notez également ici que lorsqu'on définit plusieurs animations comme cela, les animations vont par défaut toutes se lancer **en même temps** et non pas à la suite les unes des autres.

La propriété animation-duration

La propriété **animation-duration** nous permet de définir le temps que doit durer une animation. On doit préciser une durée en secondes.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-13.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1"></div>
14 <div class="d2"></div>
15 <div class="d3"></div>
16
17 </body>
18 </html>
```

```
1 div{
2 width: 100%;
3 height: 100px;
4 background-color: orange;
5 margin: 0 auto 20px auto;
6 border: 2px solid black;
7 box-sizing: border-box;
8 }
9 .d1{
10 animation-name: couleur;
11 animation-duration: 2s;
12 }
13 .d2{
14 animation-name: couleur;
15 animation-duration: 5s;
16 }
17 .d3{
18 animation-name: couleur;
19 animation-duration: 10s;
20 }
21 @keyframes couleur{
22 from{background-color: orange;}
23 50%{background-color: purple;}
24 to{background-color: orange;}
25 }
```

Ici, on fournit la même règle **@keyframes** à chacun de nos trois **div** et on définit ensuite des paramètres d'animations différents à partir de cette règle en donnant des durées d'animation différentes. Une nouvelle fois, je vous rappelle qu'une règle **@keyframes** n'est qu'un cadre qui sert qu'à définir différentes valeurs pour une propriété à différents stades d'une animation quelconque.

Ensuite, n'importe quelle animation (qui va être construite et déclarée à proprement parler par les différentes propriétés animation-*) va pouvoir se resservir de cette règle **@keyframes**. C'est la raison pour laquelle je peux ici utiliser ma règle **@keyframes** dans trois animations pour trois éléments différents.

La propriété animation-timing-function

La propriété **animation-timing-function** va nous permettre comment doit progresser l'animation entre les différentes valeurs de **keyframes** : la progression de l'animation peut être linéaire, s'accélérer de plus en plus, etc.

Nous allons pouvoir passer les valeurs suivantes à **animation-timing-function** :

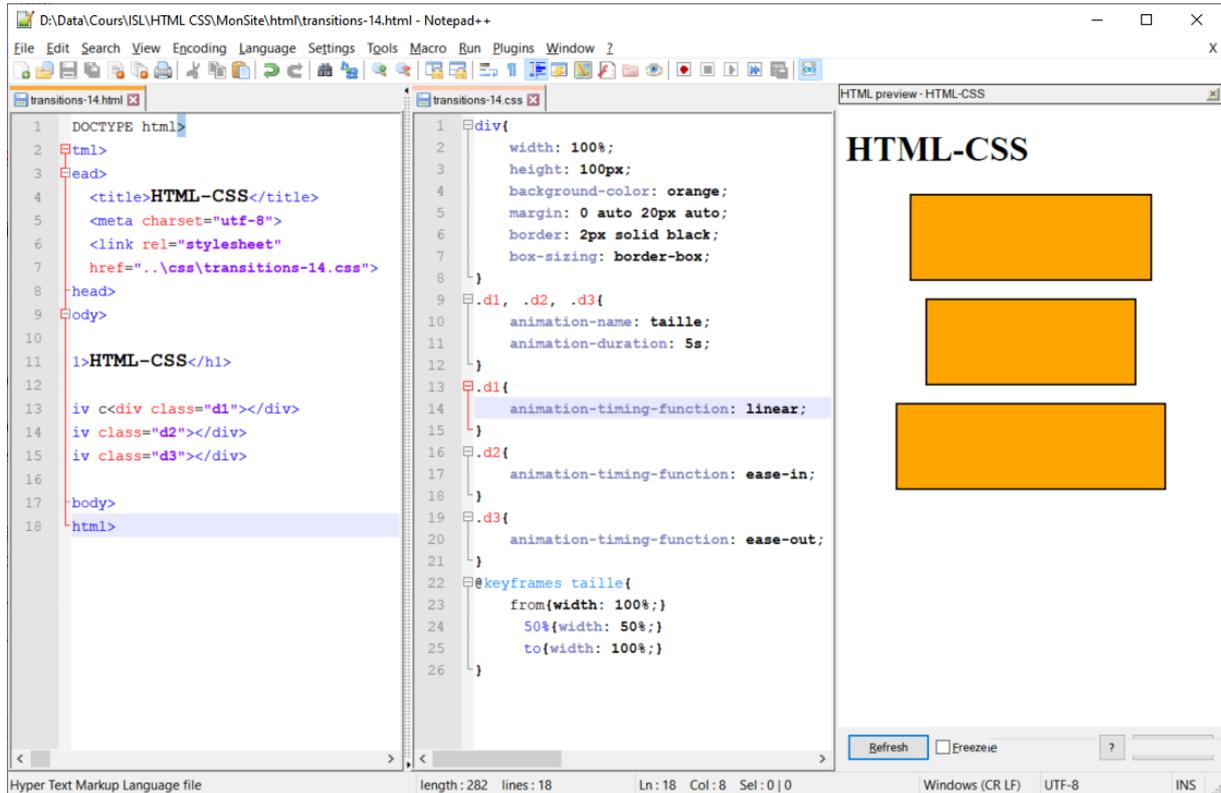
- **ease** : valeur par défaut. Entre deux valeurs de keyframes, l'animation va commencer relativement lentement, puis accélérer au milieu et se terminer lentement ;
- **linear** : Entre deux valeurs de keyframes, l'animation aura une vitesse constante du début à la fin ;
- **ease-in** : Entre deux valeurs de keyframes, l'animation va commencer lentement puis accélérer jusqu'à atteindre la prochaine valeur de keyframe ;
- **ease-out** : Entre deux valeurs de keyframes, l'animation va commencer rapidement et décélérer progressivement jusqu'à atteindre la prochaine valeur de keyframe ;
- **ease-in-out** : Entre deux valeurs de keyframes, l'animation commence lentement, accélère au milieu et finit lentement ;
- **cubic-bezier(x1, y1, x2, y2)** : permet de définir une courbe de Bézier spécifique pour créer une animation à la vitesse totalement contrôlée.

Notre but n'est ici bien évidemment pas de faire un cours sur les courbes de Bézier. Vous pouvez simplement retenir les équivalents mot-clef/Bézier suivants qui peuvent se révéler utiles :

- **transition-timing-function : ease** est équivalent à **transition-timing-function : cubic-bezier(0.25, 0.1, 0.25, 1)** ;
- **transition-timing-function : ease-in** est équivalent à **transition-timing-function : cubic-bezier(0.42, 0, 1, 1)** ;
- **transition-timing-function : ease-out** est équivalent à **transition-timing-function : cubic-bezier(0, 0, 0.58, 1)** ;
- **transition-timing-function : ease-in-out** est équivalent à **transition-timing-function : cubic-bezier(0.42, 0, 0.58, 1)**.

Concernant la propriété **animation-timing-function**, vous devez bien comprendre que les valeurs passées vont dicter le comportement des animations entre chaque valeur de **keyframes**, c'est-à-dire que le pattern donné par **animation-timing-function** va être appliqué et répété entre chaque valeur de **keyframes** et ne va pas définir l'animation au complet.

Par exemple, si on définit une règle **@keyframes** avec une valeur intermédiaire à 50% et que l'animation se fait selon **animation-timing-function: ease**, l'animation va commencer relativement lentement à partir du **from**, puis accélérer pour arriver lentement au **50%** puis repartir relativement lentement à partir du 50% pour accélérer à nouveau et finir à nouveau lentement au niveau du **to**.



```

1 DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-14.css">
8 </head>
9 <body>
10 <h1>HTML-CSS</h1>
11 <div class="d1"></div>
12 <div class="d2"></div>
13 <div class="d3"></div>
14 </body>
15 </html>

```

```

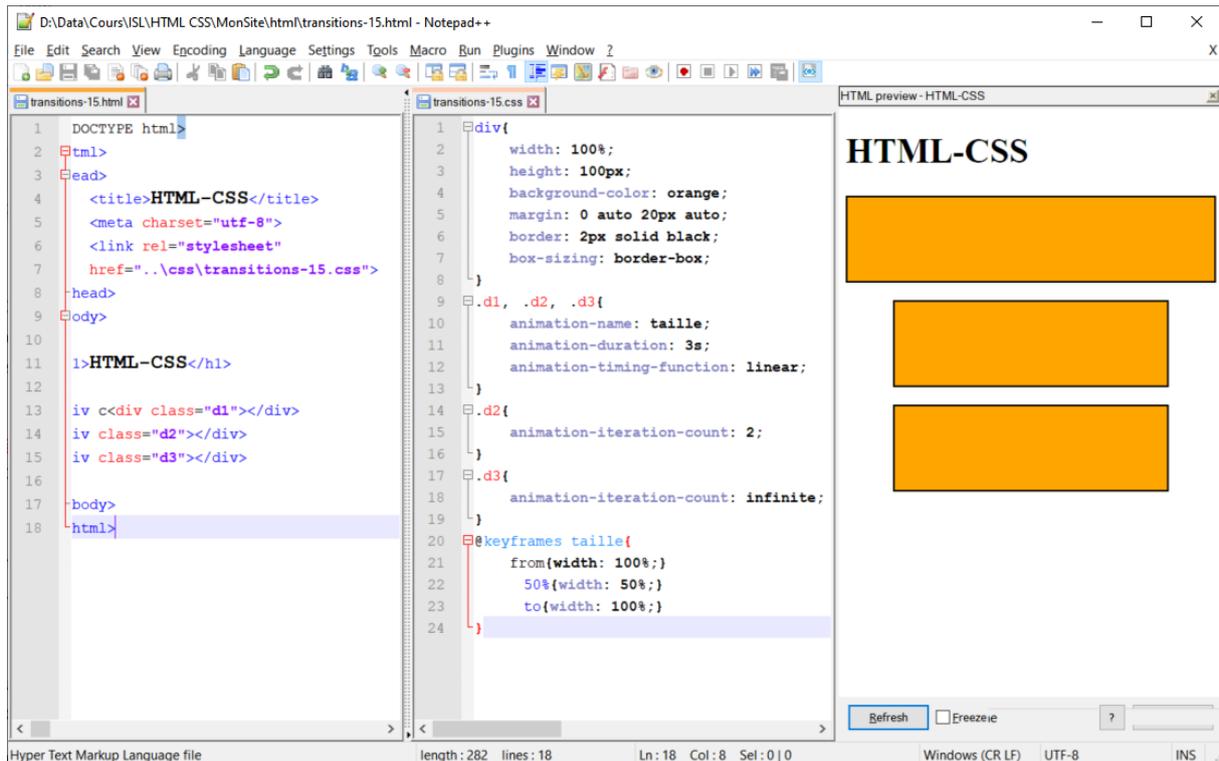
1 div{
2 width: 100%;
3 height: 100px;
4 background-color: orange;
5 margin: 0 auto 20px auto;
6 border: 2px solid black;
7 box-sizing: border-box;
8 }
9 .d1, .d2, .d3{
10 animation-name: taille;
11 animation-duration: 5s;
12 }
13 .d1{
14 animation-timing-function: linear;
15 }
16 .d2{
17 animation-timing-function: ease-in;
18 }
19 .d3{
20 animation-timing-function: ease-out;
21 }
22 @keyframes taille{
23 from{width: 100%;}
24 50%{width: 50%;}
25 to{width: 100%;}
26 }

```

La propriété animation-iteration-count

La propriété **animation-iteration-count** va nous permettre de définir combien de fois une animation va être jouée. Par défaut, une animation ne sera jouée qu'une fois.

Pour modifier ce comportement par défaut, on va pouvoir passer soit un nombre à **animation-iteration-count** qui va correspondre au nombre de fois que l'on souhaite jouer l'animation, soit le mot clef **infinite** qui signifie que l'animation va se répéter à l'infini.



```

1 DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-15.css">
8 </head>
9 <body>
10 <h1>HTML-CSS</h1>
11 <div class="d1"></div>
12 <div class="d2"></div>
13 <div class="d3"></div>
14 </body>
15 </html>

```

```

1 div{
2 width: 100%;
3 height: 100px;
4 background-color: orange;
5 margin: 0 auto 20px auto;
6 border: 2px solid black;
7 box-sizing: border-box;
8 }
9 .d1, .d2, .d3{
10 animation-name: taille;
11 animation-duration: 3s;
12 animation-timing-function: linear;
13 }
14 .d2{
15 animation-iteration-count: 2;
16 }
17 .d3{
18 animation-iteration-count: infinite;
19 }
20 @keyframes taille{
21 from{width: 100%;}
22 50%{width: 50%;}
23 to{width: 100%;}
24 }

```

Ici, ma première animation (l'animation correspondante à mon `div class="d1"`) ne va être jouée qu'une fois par défaut.

Ensuite, on demande à notre deuxième animation de se répéter, c'est-à-dire d'être jouée deux fois avec `animation-iteration-count: 2`.

Finalement, notre troisième animation va se répéter à l'infini grâce à `animation-iteration-count: infinite`.

La propriété `animation-direction`

La propriété `animation-direction` va nous permettre de spécifier le sens dans lequel une animation doit être jouée, c'est-à-dire si elle doit être jouée en partant du début ou de la fin pour une ou plusieurs de ses itérations ou répétitions.

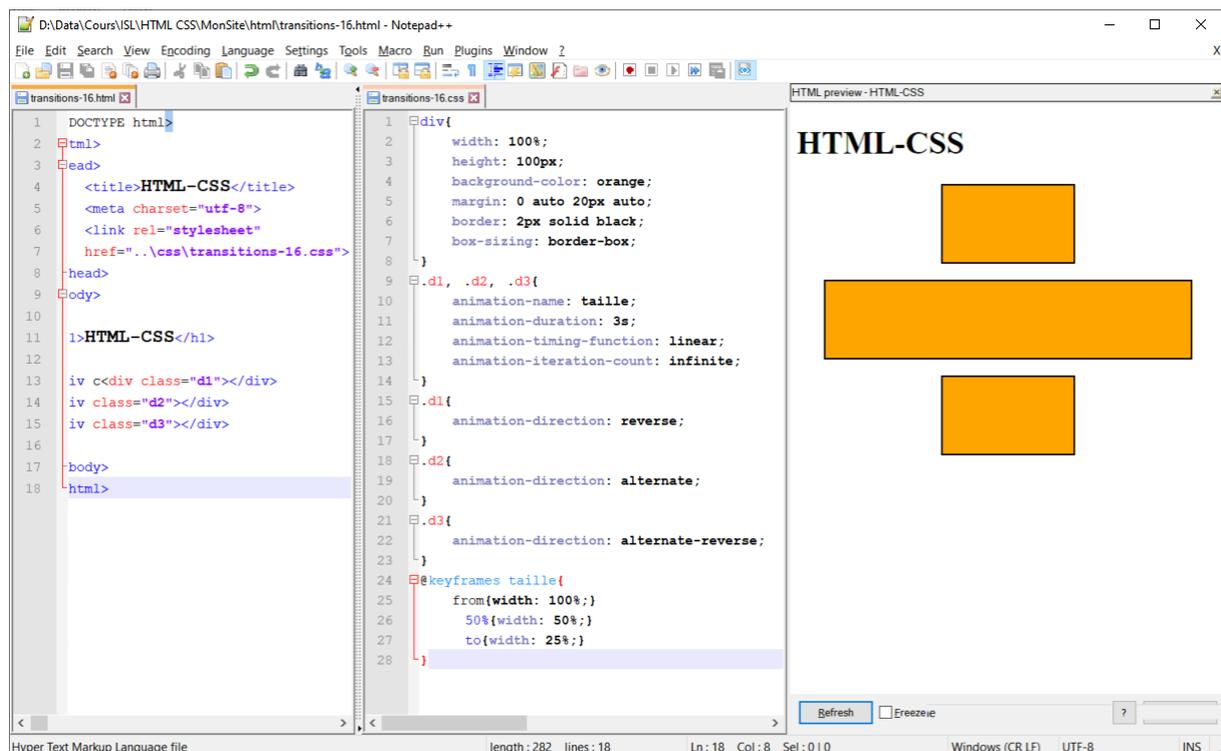
Nous allons pouvoir passer les valeurs suivantes à `animation-direction` :

- **normal** : valeur par défaut. L'animation est jouée dans le sens dans lequel elle a été déclarée (du from vers le to) ;
- **reverse** : l'animation est jouée dans le sens inverse pour toutes ses itérations ;
- **alternate** : l'animation va être jouée une première fois dans le sens normal, puis dans le sens contraire, puis à nouveau dans le sens normal et etc. ;

- **alternate-reverse** : l'animation va être jouée une première fois dans le sens inverse, puis dans le sens normal, puis à nouveau dans le sens inverse et etc..

Notez qu'en inversant le sens d'une animation pour une ou plusieurs de ses itérations, les propriétés liées au timing seront également inversées. Par exemple, une animation possédant une **animation-direction : reverse** et une **animation-timing-function : ease-in** sera jouée comme si la valeur était **animation-timing-function : ease-out**.

Notez également que les valeurs **animation-direction : alternate** et **animation-direction : alternate-reverse** ne vont évidemment n'avoir un effet que pour les animations qui vont être jouées plus d'une fois.



```

1 DOCTYPE html
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-16.css">
8 </head>
9 <body>
10 <h1>HTML-CSS</h1>
11 <div class="d1"></div>
12 <div class="d2"></div>
13 <div class="d3"></div>
14 </body>
15 </html>
16
17 @keyframes taille{
18 from{width: 100%;}
19 50%{width: 50%;}
20 to{width: 25%;}
21
22 .d1{
23 animation-name: taille;
24 animation-duration: 3s;
25 animation-timing-function: linear;
26 animation-iteration-count: infinite;
27 animation-direction: reverse;
28
29 .d2{
30 animation-direction: alternate;
31
32 .d3{
33 animation-direction: alternate-reverse;
34
35

```

Ici, nous créons une règle **@keyframes** qui va modifier la valeur de la propriété **width** de 100% vers 25%. Dans son sens normal, l'animation fait donc passer nos **div** d'une taille de 100% à 25%.

Ensuite, nous demandons à chacune de nos animations de se répéter à l'infini pour bien voir le comportement lié aux valeurs **alternate** et **alternate-reverse**.

Pour notre première animation, cependant, on indique une propriété **animation-direction : reverse** qui signifie qu'on souhaite que l'animation joue dans le sens contraire. Le départ de notre animation va donc utiliser la taille **width: 25%** et la taille du **div** va grandir jusqu'à 100%.

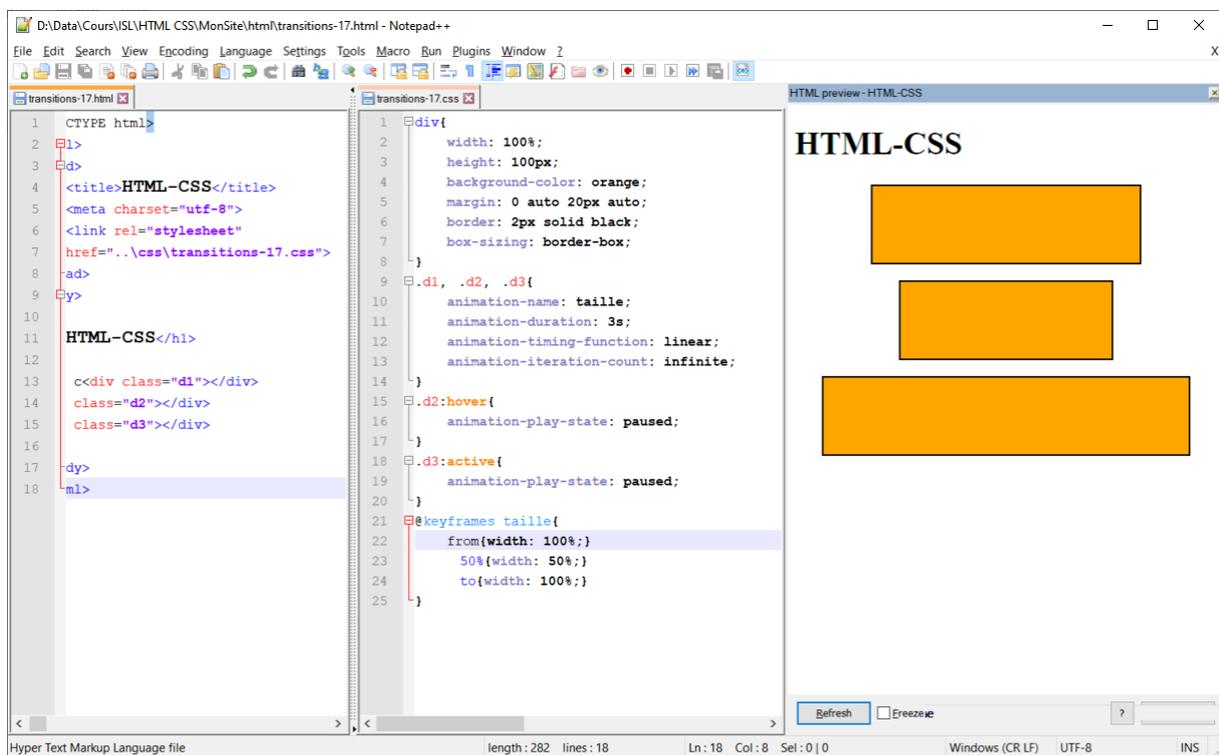
Pour notre deuxième animation, on passe la valeur **alternate** à la propriété **animation-direction**. Notre animation va donc jouer une première fois dans le sens normal, puis dans le sens contraire, puis à nouveau dans le sens normal et etc.

La dernière animation va avoir le comportement opposé de la deuxième : elle va commencer à jouer dans le sens inverse puis alterner à chaque nouvelle itération (ou répétition) puisqu'on lui a donné une **animation-direction : alternate-reverse**

La propriété animation-play-state

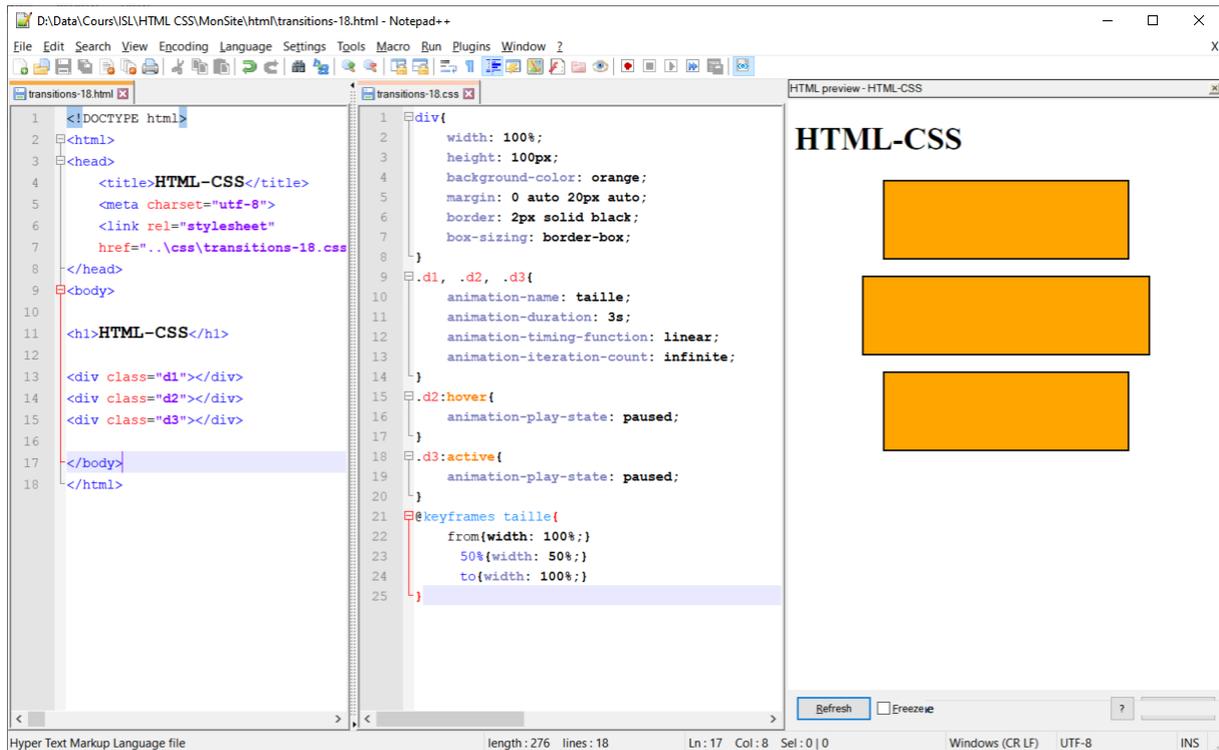
La **propriété animation-play-state** va nous permettre de définir si une animation doit être jouée ou être en pause. On va pouvoir lui passer la valeur **running** (l'animation s'exécute normalement) ou **paused** (l'animation est mise en pause).

Cette propriété va pouvoir être utile pour mettre une animation en pause à un certain point de l'animation ou selon une certaine action de l'utilisateur. Par exemple, on va pouvoir proposer aux utilisateurs de mettre en pause une animation lorsqu'ils passent le curseur de leur souris sur l'élément pour lequel une animation est jouée ou lorsqu'ils cliquent (en gardant le clic enfoncé) sur l'élément en utilisant les pseudo classes **:hover** et **:active**.



```

1  CTYPE html
2  <!DOCTYPE html>
3  <html>
4  <title>HTML-CSS</title>
5  <meta charset="utf-8">
6  <link rel="stylesheet"
7  href=".css/transitions-17.css">
8  <body>
9  <div class="d1"></div>
10 <div class="d2"></div>
11 <div class="d3"></div>
12 </body>
13 </html>
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2
```



La propriété animation-fill-mode

Par défaut, une règle **@keyframes** ou plus généralement une animation ne va pas affecter ni donc définir la valeur de la propriété animée en dehors du temps de l'animation.

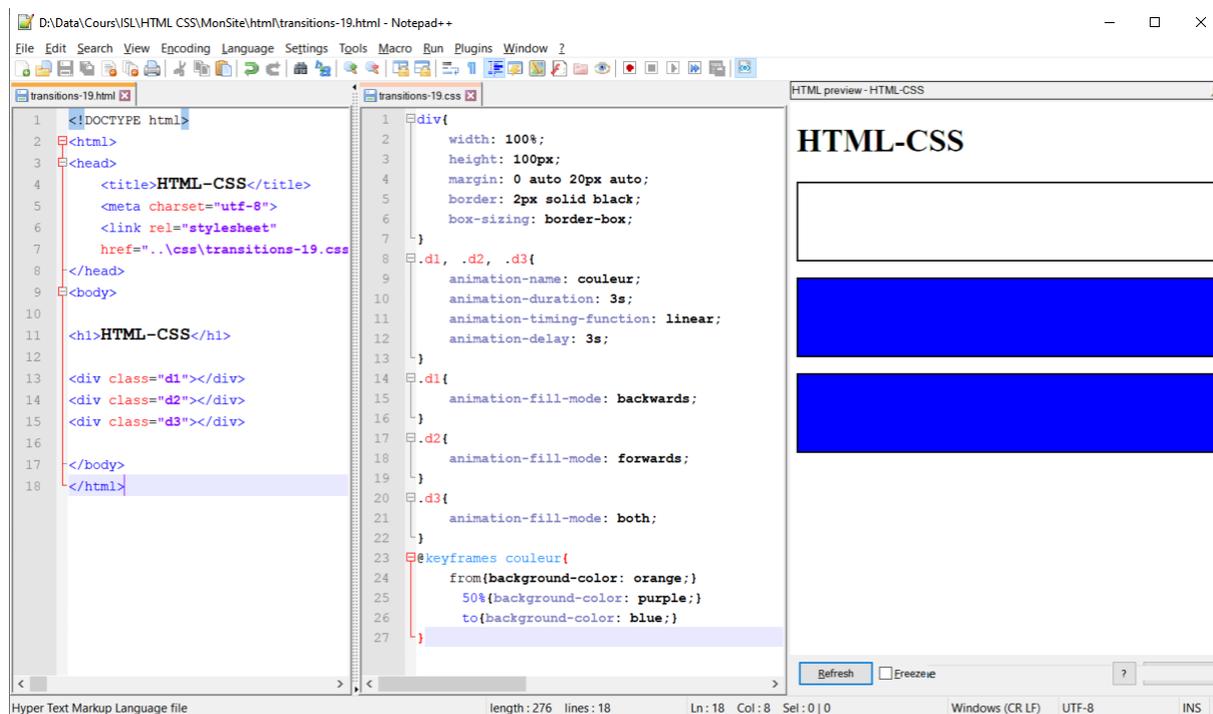
Nous avons vu cela en début de tutoriel et c'est la raison pour laquelle je définis depuis le début de celui-ci un comportement « normal » ou par défaut pour les propriétés animées en CSS.

Cependant, il existe un moyen plus élégant de faire cela en utilisant la propriété **animation-fill-mode**. Cette propriété va en effet nous permettre conserver le comportement de nos propriétés défini dans l'animation en dehors de l'animation (avant ou après).

On va ainsi pouvoir appliquer la valeur de départ de notre animation à notre propriété avant que l'animation ne commence ou la valeur d'arrivée de notre animation à la propriété après que celle-ci soit finie.

Pour faire cela, nous allons pouvoir passer l'une des valeurs suivantes à **animation-fill-mode** :

- **backwards** : notre propriété animée utilisera la valeur de départ de l'animation comme valeur avant que l'animation ne commence ;
- **forwards** : notre propriété animée utilisera la valeur de fin de l'animation comme valeur après que l'animation soit terminée ;
- **both** : notre propriété animée utilisera la valeur de départ de l'animation comme valeur avant que l'animation ne commence et la valeur de fin de l'animation comme valeur après que l'animation soit terminée.



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-19.css"
8 </head>
9 <body>
10 <h1>HTML-CSS</h1>
11 <div class="d1"></div>
12 <div class="d2"></div>
13 <div class="d3"></div>
14 </body>
15 </html>
16
17
18
19
20
21
22
23
24
25
26
27

```

Dans l'exemple ci-dessus, j'applique un délai de 3 secondes à chacune de mes animations.

La première animation a un **animation-fill-mode : backwards**, elle utilisera donc la valeur de départ de l'animation durant le délai tout comme la dernière animation qui possède un **animation-fill-mode : both**.

Pour la deuxième animation, au contraire, la propriété n'utilisera pas de valeur avant le début de l'animation puisque rien ne lui a été précisé.

Après la fin de l'animation, c'est l'inverse qui va se passer : notre deuxième animation va conserver la valeur de fin de l'animation de la propriété grâce à **animation-fill-mode : forwards** tandis que notre première animation n'aura plus aucune valeur attachée à son **background-color**.

La propriété animation

La propriété **animation** correspond à la version raccourcie ou notation short-hand des propriétés **animation-*** vues ci-dessus.

Nous allons pouvoir lui passer les différentes valeurs relatives aux propriétés animation- pour créer simplement des animations complètes. Il est généralement considéré comme une bonne pratique de passer les valeurs dans l'ordre suivant pour être certain que l'animation fonctionne correctement :

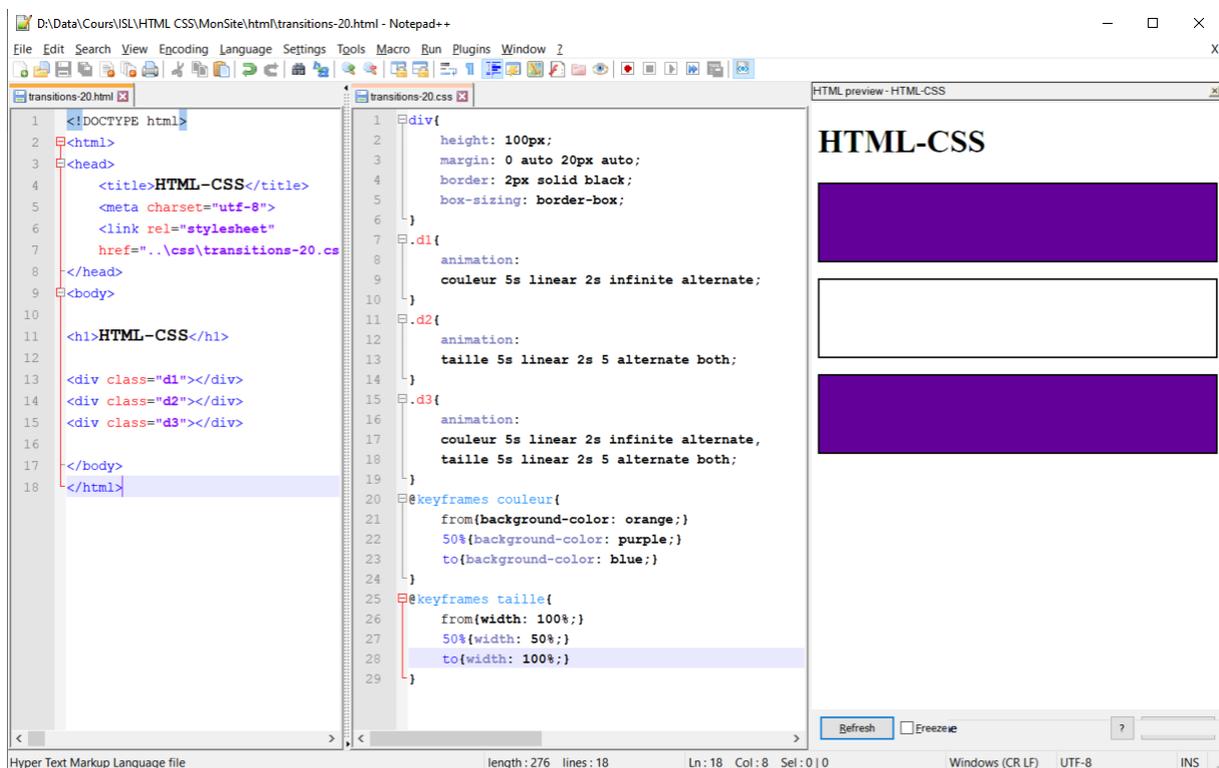
1. La valeur relative à la propriété **animation-name** ;
2. La valeur relative à la propriété **animation-duration** ;
3. La valeur relative à la propriété **animation-timing-function** ;

4. La valeur relative à la propriété **animation-delay** ;
5. La valeur relative à la propriété **animation-iteration-count** ;
6. La valeur relative à la propriété **animation-direction** ;
7. La valeur relative à la propriété **animation-fill-mode** ;
8. La valeur relative à la propriété **animation-play-state** .

Bien évidemment, nous n'allons pas être obligé de préciser toutes les valeurs pour chaque animation : si une valeur est omise, la valeur par défaut de la propriété correspondante sera utilisée.

Cependant, notez bien que la première valeur de type « secondes » fournie à animation sera toujours considérée comme étant la durée de l'animation tandis que la deuxième sera toujours considérée comme le délai.

On va également pouvoir créer plusieurs animations avec la propriété animation. Pour cela, il suffira de séparer les déclarations des différentes animations par une virgule.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-20.css"
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1"></div>
14 <div class="d2"></div>
15 <div class="d3"></div>
16
17 </body>
18 </html>
19
20 @keyframes couleur{
21 from(background-color: orange;)
22 50%(background-color: purple;)
23 to(background-color: blue;)
24 }
25 @keyframes taille{
26 from(width: 100%;)
27 50%(width: 50%;)
28 to(width: 100%;)
29 }
30
31 .div{
32 height: 100px;
33 margin: 0 auto 20px auto;
34 border: 2px solid black;
35 box-sizing: border-box;
36 }
37 .d1{
38 animation:
39 couleur 5s linear 2s infinite alternate;
40 }
41 .d2{
42 animation:
43 taille 5s linear 2s 5 alternate both;
44 }
45 .d3{
46 animation:
47 couleur 5s linear 2s infinite alternate,
48 taille 5s linear 2s 5 alternate both;
49 }
```

Ici, dans notre premier exemple, on anime la couleur de fond de notre élément **div** sur une durée de 5 secondes. L'animation doit progresser de façon linéaire et commencer après un délai de 2 secondes. De plus, on demande à l'animation de se répéter à l'infini et d'alterner le sens dans lequel elle est jouée à chaque fois.

Dans notre deuxième exemple, on anime la taille du **div** sur 5 secondes. L'animation doit à nouveau progresser de façon linéaire et commencer après un délai de 2 secondes. L'animation doit se répéter

5 fois et nous demandons à ce que la propriété utilise la valeur de départ de l'animation avant que celle-ci ne commence et la valeur d'arrivée après la fin de l'animation.

Notre troisième exemple fait jouer les deux animations précédentes en même temps. Pour cela, on sépare les différentes déclarations par une virgule dans `animation`, tout simplement.

Créer des transformations en CSS

Dans les leçons précédentes, nous avons pu étudier les transitions et les animations en CSS qui permettent de modifier la valeur de certaines propriétés progressivement en fonction de certains critères (chargement de la page, passage de souris sur l'élément, etc.) et donc d'ajouter un côté interactif à nos pages web.

Le CSS va également nous permettre d'appliquer des transformations à nos éléments : on va pouvoir incliner nos éléments, les déformer, les translater, etc.

Dans cette leçon, nous allons expliquer en détail comment fonctionnent les transformations et apprendre à créer des transformations plus ou moins complexes.

Définir une transformation en CSS

La possibilité d'effectuer des transformations en CSS est récente et les possibilités et fonctionnalités des transformations sont donc aujourd'hui relativement limitées.

Cependant, on peut s'attendre à ce que de nouvelles fonctionnalités viennent s'ajouter aux transformations dans un futur proche.

A terme, les transformations en CSS devraient être définies par 3 critères qui vont pouvoir être renseignés via 3 propriétés CSS différentes :

- La propriété **transform-box** va nous permettre de définir une boîte de référence qui va être utilisée pour calculer le point d'origine et pour réaliser la transformation en soi ;
- La propriété **transform-origin** va nous permettre de définir un point d'origine à partir duquel réaliser la transformation ;
- La propriété **transform** va nous permettre de définir un effet de transformation.

La propriété **transform-box** ne fait pas encore partie des recommandations du W3C et est toujours en développement. Il est donc déconseillé de l'utiliser pour l'instant puisque sa définition n'est pas encore stable et que le support par les navigateurs n'est pas assuré. Nous ne l'étudierons ainsi pas ici.

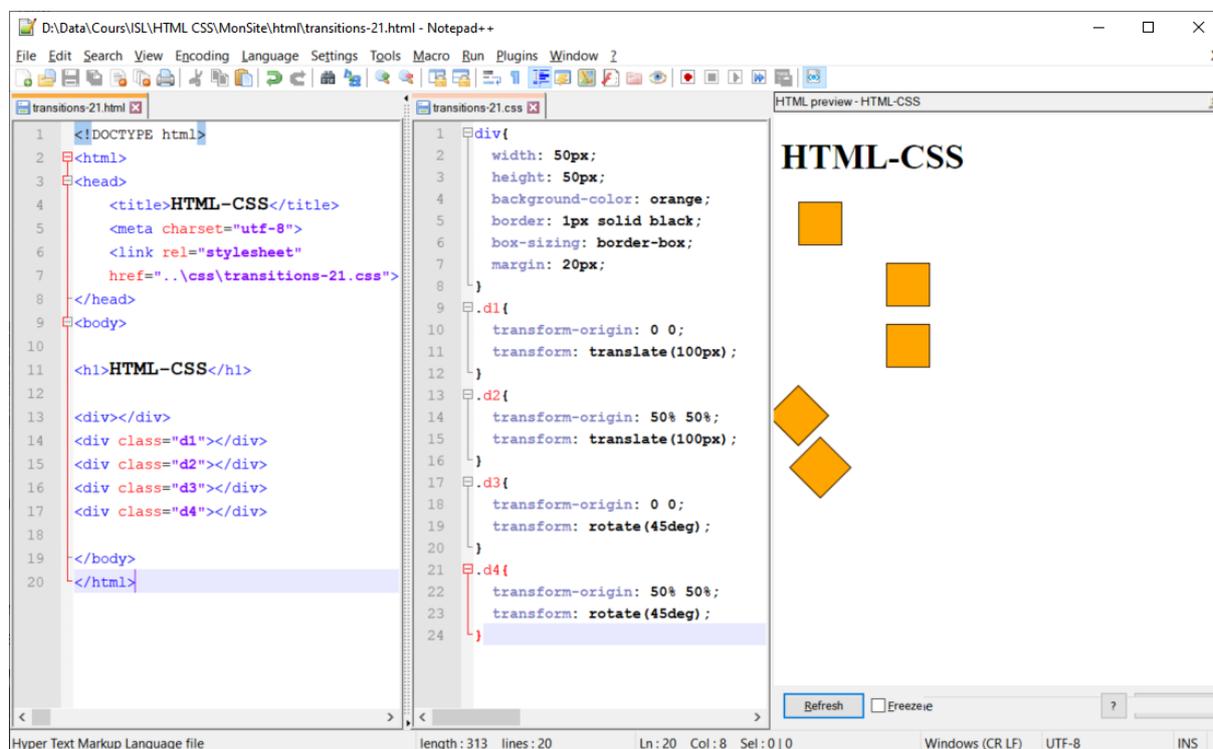
La propriété **transform-origin** permet de définir un point d'origine pour une transformation. Par défaut, le point d'origine est le centre de l'élément.

Cette propriété va pouvoir prendre une ou deux valeurs. En n'indiquant qu'une valeur, la valeur sera utilisée pour définir les coordonnées dans l'axe horizontal et dans l'axe vertical du point d'origine. En indiquant deux valeurs, la première valeur va permettre de définir la coordonnée horizontale du point d'origine tandis que la seconde valeur va permettre de définir sa coordonnée verticale.

Nous allons pouvoir passer les mots clefs **top**, **right**, **bottom**, **left** et **center** ainsi que des longueurs ou des pourcentages à cette propriété. Les valeurs de type longueur ou pourcentage vont définir l'éloignement du point d'origine à partir du bord supérieur gauche de la boîte de référence.

La propriété **transform** va nous permettre de définir un ou plusieurs effets de transformation à appliquer à un élément : inclinaison, rotation, déformation, etc.

Cette propriété va accepter différents mots clefs ou plus exactement différentes fonctions qui vont définir le type de transformation qui va être appliqué à un élément. Nous allons étudier ces valeurs dans la suite de la leçon.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-21.css">
8 </head>
9 <body>
10 <h1>HTML-CSS</h1>
11 <div></div>
12 <div class="d1"></div>
13 <div class="d2"></div>
14 <div class="d3"></div>
15 <div class="d4"></div>
16 </body>
17 </html>
```

```
1 div{
2 width: 50px;
3 height: 50px;
4 background-color: orange;
5 border: 1px solid black;
6 box-sizing: border-box;
7 margin: 20px;
8 }
9 .d1{
10 transform-origin: 0 0;
11 transform: translate(100px);
12 }
13 .d2{
14 transform-origin: 50% 50%;
15 transform: translate(100px);
16 }
17 .d3{
18 transform-origin: 0 0;
19 transform: rotate(45deg);
20 }
21 .d4{
22 transform-origin: 50% 50%;
23 transform: rotate(45deg);
24 }
```

Dans le code ci-dessus, nous avons défini 4 transformations pour nos 4 **div** portant des attributs class.

Nous appliquons un premier effet de transformation à nos deux premiers **div** qui va être une translation, c'est-à-dire un déplacement selon une direction. Ici, on va donc déplacer nos deux **div** de 100px à partir de leur point d'origine vers la droite.

Ensuite, nous appliquons un effet de rotation à nos deux derniers **div**. La rotation va s'effectuer dans le sens des aiguilles d'une montre.

Le point d'origine de la transformation pour le premier **div** est son coin supérieur gauche tandis que cela va être le centre pour le deuxième. Comme la transformation n'est ici qu'un déplacement horizontal, modifier le point d'origine ne change pas le résultat de la transformation.

En revanche, cela va être différent pour une rotation : notre troisième **div** va pivoter autour d'un point central qui va être son coin supérieur gauche tandis que notre quatrième **div** va pivoter autour de son point central. Dans ce cas-là, modifier le point d'origine de la transformation change le résultat obtenu.

Finalement, vous pouvez noter ici qu'à la différence des transitions et des animations pour lesquelles on précise une durée, les transformations sont permanentes et les éléments vont conserver la transformation.

Exemples de transformations 2D

Nous allons pour le moment nous concentrer sur les effets de transformations en deux dimensions. En effet, vous devez savoir qu'on peut également aujourd'hui créer des transformations en 3D en rajoutant une perspective à nos éléments. Nous discuterons de cette possibilité plus tard.

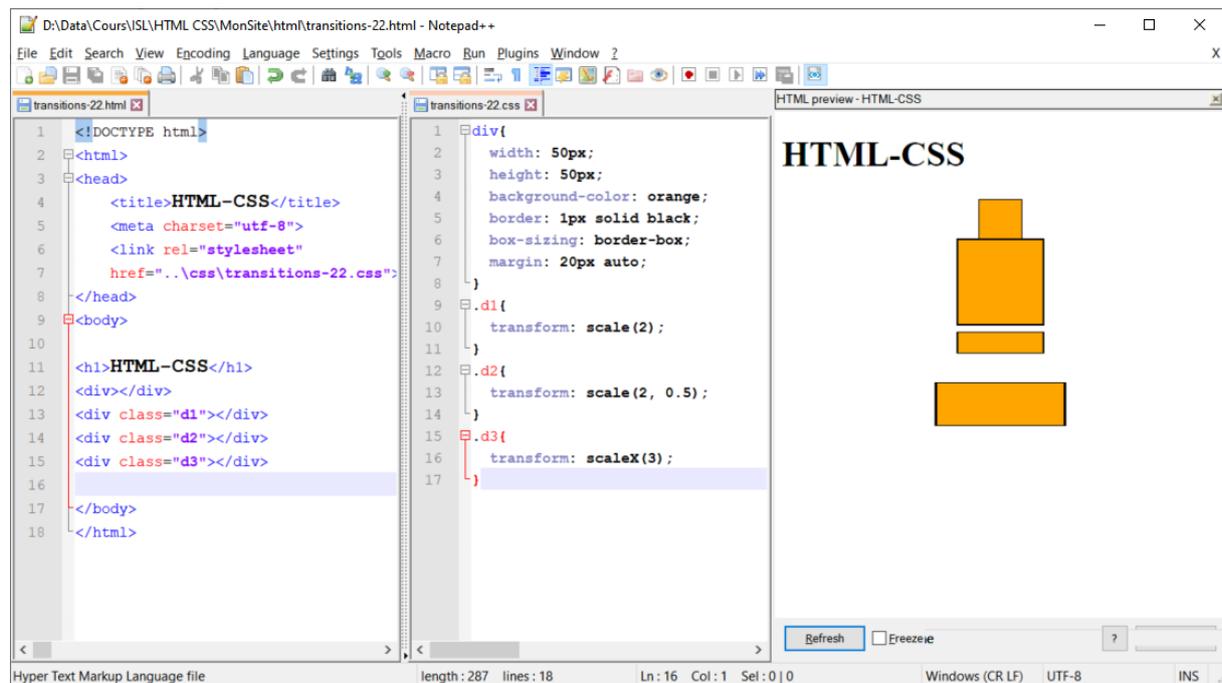
Modifier la taille ou l'échelle d'un élément avec `scale()`

La fonction `scale()` permet de modifier la taille ou plus exactement l'échelle de l'élément. Nous allons pouvoir lui passer deux nombres qui vont correspondre au pourcentage d'agrandissement en largeur et en hauteur de l'élément.

Par exemple, en écrivant `transform : scale(2, 0.5)`, l'élément va doubler en largeur et être diminué de moitié en hauteur.

Notez que cette fonction ne va pas affecter les propriétés `width` et `height` de l'élément mais également s'appliquer au `font-size` et au `padding`.

Finalement, vous devez savoir que la fonction `scale()` est la notation raccourcie des fonctions `scaleX()` et `scaleY()` qui permettent de ne modifier que la largeur et la hauteur respectivement d'un élément.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-22.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12 <div></div>
13 <div class="d1"></div>
14 <div class="d2"></div>
15 <div class="d3"></div>
16
17 </body>
18 </html>
```

```
1 div{
2 width: 50px;
3 height: 50px;
4 background-color: orange;
5 border: 1px solid black;
6 box-sizing: border-box;
7 margin: 20px auto;
8 }
9 .d1{
10 transform: scale(2);
11 }
12 .d2{
13 transform: scale(2, 0.5);
14 }
15 .d3{
16 transform: scaleX(3);
17 }
```

HTML-CSS

Refresh Freeze

length: 287 lines: 18 Ln: 16 Col: 1 Sel: 0 | 0 Windows (CR LF) UTF-8 INS

Ici, on ne passe qu'une valeur à notre fonction `scale()` pour notre première transformation : cette valeur va donc être utilisée pour calculer la mise à l'échelle horizontale et verticale et notre div fera deux fois sa taille d'origine.

Pour notre deuxième transformation, on demande à ce que le div devienne deux fois plus large et qu'il soit deux fois moins haut.

Finalement, on utilise la fonction `scaleX()` pour définir notre dernière transformation qui ne va donc impacter que la largeur de l'élément.

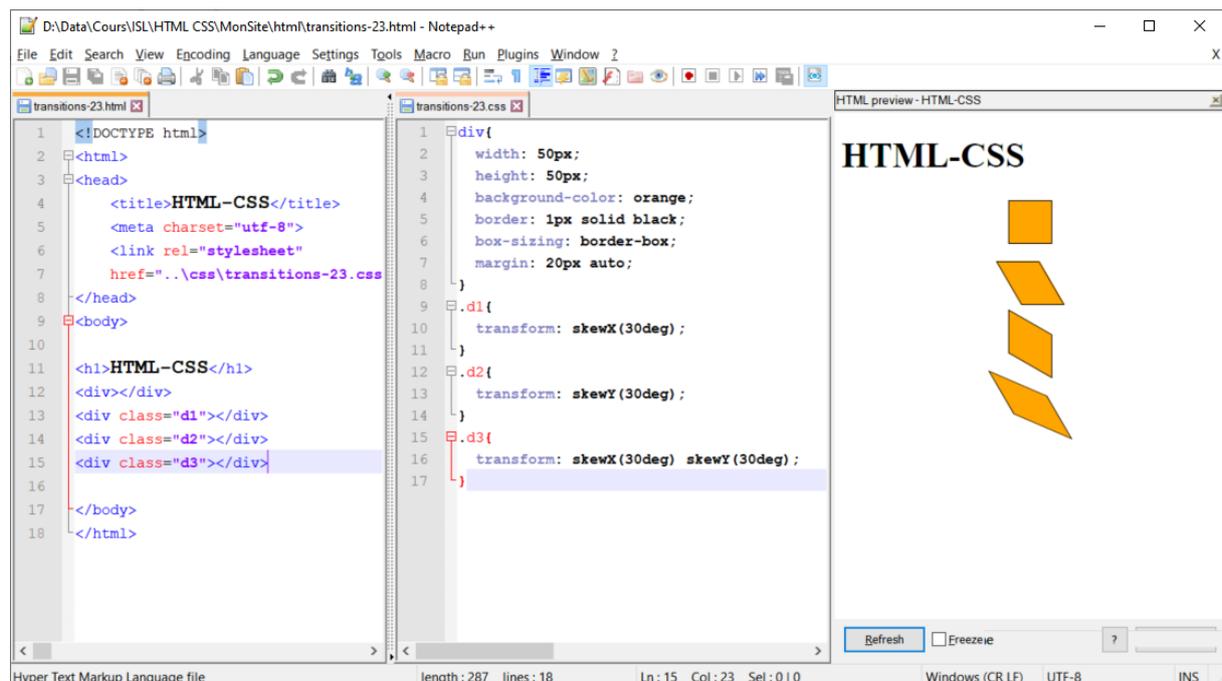
Notez bien ici qu'appliquer une transformation à des éléments ne va pas modifier la taille de l'espace qui leur était attribué à la base ni donc faire bouger les autres éléments autour en conséquence. Il faudra donc faire bien attention à ce que nos éléments transformés chevauchent pas leurs voisins et ne dépassent pas de leur parent.

Déformer un élément avec `skewX()` et `skewY()`

Avant toute chose, il faut savoir que la fonction `skew()`, si elle existe, n'a été créée que pour des questions de compatibilité et qu'on ne devrait jamais l'utiliser.

A la place, il faudra plutôt utiliser les fonctions `skewX()` et `skewY()` qui vont nous permettre de déformer un élément selon son axe horizontal ou vertical.

Nous allons devoir passer un angle (généralement en `deg`) à ces deux fonctions qui va représenter l'angle selon lequel l'élément doit être déformé le long de l'axe correspondant.



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>HTML-CSS</title>
5   <meta charset="utf-8">
6   <link rel="stylesheet"
7     href="..\css\transitions-23.css"
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12 <div></div>
13 <div class="d1"></div>
14 <div class="d2"></div>
15 <div class="d3"></div>
16
17 </body>
18 </html>
  
```

```

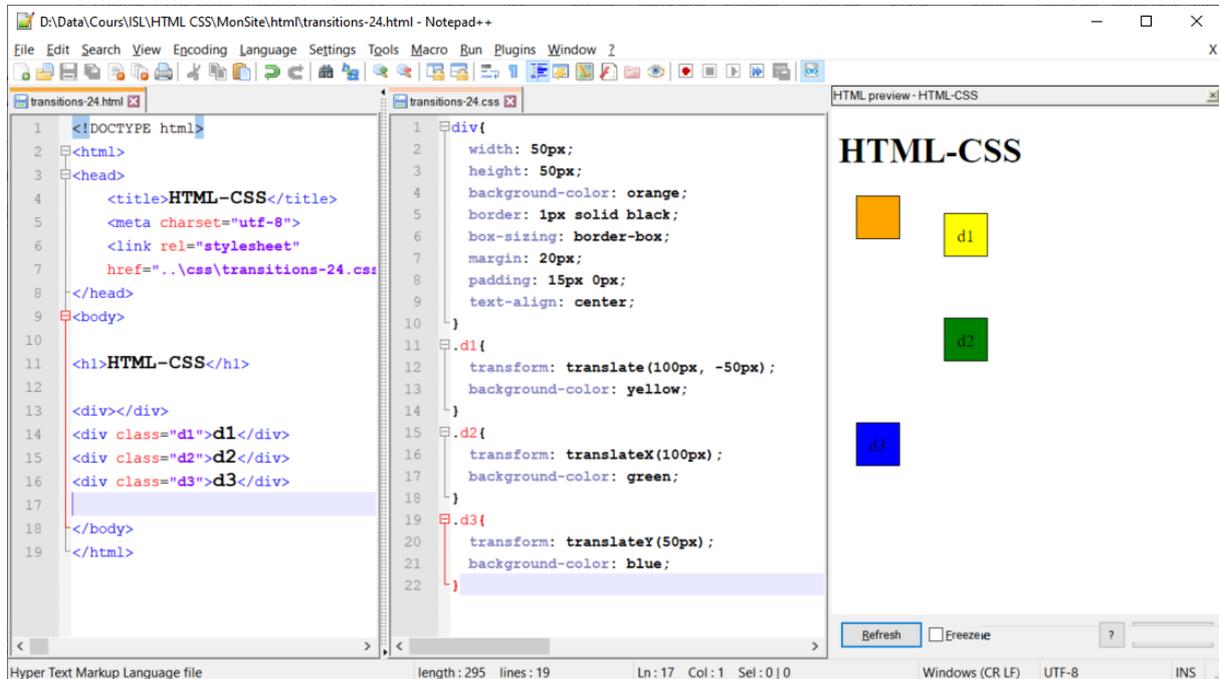
1 div{
2   width: 50px;
3   height: 50px;
4   background-color: orange;
5   border: 1px solid black;
6   box-sizing: border-box;
7   margin: 20px auto;
8 }
9
10 .d1{
11   transform: skewX(30deg);
12 }
13
14 .d2{
15   transform: skewY(30deg);
16 }
17
18 .d3{
19   transform: skewX(30deg) skewY(30deg);
20 }
  
```

Effectuer une translation avec `translate(X,Y)`

La fonction `translate()` va nous permettre de créer une translation, c'est-à-dire de déplacer un élément selon un certain vecteur (ou selon une certaine distance et direction, pour faire simple).

Là encore, on va pouvoir passer deux valeurs de type longueur à `translate()` pour spécifier les caractéristiques de la translation dans l'axe horizontal et dans l'axe vertical ou utiliser les versions complètes `translateX()` et `translateY()`.

Les valeurs passées vont pouvoir être positives ou négatives. Une valeur positive pour l'axe horizontal va déplacer l'élément vers la droite tandis qu'une valeur positive pour l'axe vertical va déplacer l'élément vers le bas et inversement pour des valeurs négatives.



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>HTML-CSS</title>
5   <meta charset="utf-8">
6   <link rel="stylesheet"
7     href="..\css\transitions-24.css"
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div></div>
14 <div class="d1">d1</div>
15 <div class="d2">d2</div>
16 <div class="d3">d3</div>
17
18 </body>
19 </html>

```

```

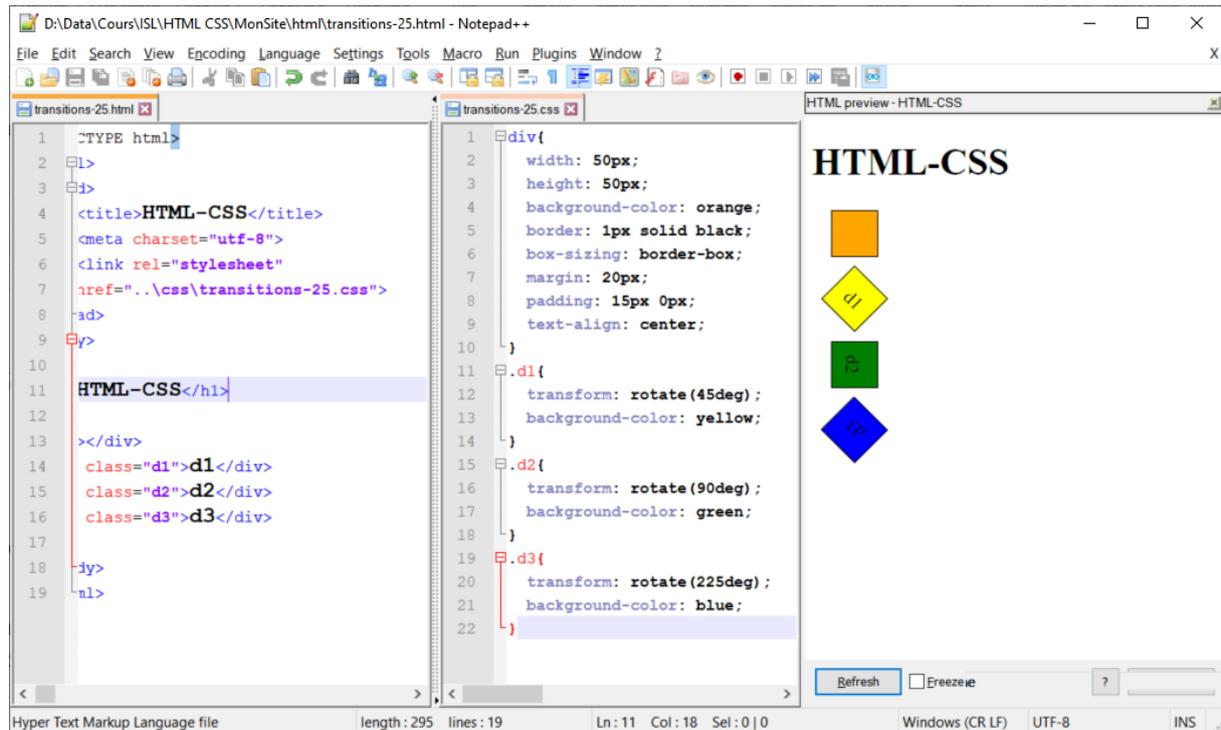
1 div{
2   width: 50px;
3   height: 50px;
4   background-color: orange;
5   border: 1px solid black;
6   box-sizing: border-box;
7   margin: 20px;
8   padding: 15px 0px;
9   text-align: center;
10 }
11 .d1{
12   transform: translate(100px, -50px);
13   background-color: yellow;
14 }
15 .d2{
16   transform: translateX(100px);
17   background-color: green;
18 }
19 .d3{
20   transform: translateY(50px);
21   background-color: blue;
22 }

```

Effectuer une rotation avec `rotate()`

La fonction `rotate()` va nous permettre de faire pivoter un élément ou de lui faire effectuer une rotation selon un certain angle. Nous allons pouvoir lui fournir un angle (généralement en `deg`) en valeur.

La rotation va s'effectuer dans le sens des aiguilles d'une montre. Ainsi, indiquer `rotate(90deg)` va faire pivoter l'élément d'un quart de tour.



```

1 <!DOCTYPE html>
2 <html>
3 </html>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7   href="..\css\transitions-25.css">
8 </link>
9 </head>
10 <body>
11 <h1>HTML-CSS</h1>
12 </h1>
13 </div>
14 <div class="d1">d1</div>
15 <div class="d2">d2</div>
16 <div class="d3">d3</div>
17 </div>
18 </body>
19 </html>

```

```

1 div{
2   width: 50px;
3   height: 50px;
4   background-color: orange;
5   border: 1px solid black;
6   box-sizing: border-box;
7   margin: 20px;
8   padding: 15px 0px;
9   text-align: center;
10 }
11 .d1{
12   transform: rotate(45deg);
13   background-color: yellow;
14 }
15 .d2{
16   transform: rotate(90deg);
17   background-color: green;
18 }
19 .d3{
20   transform: rotate(225deg);
21   background-color: blue;
22 }

```

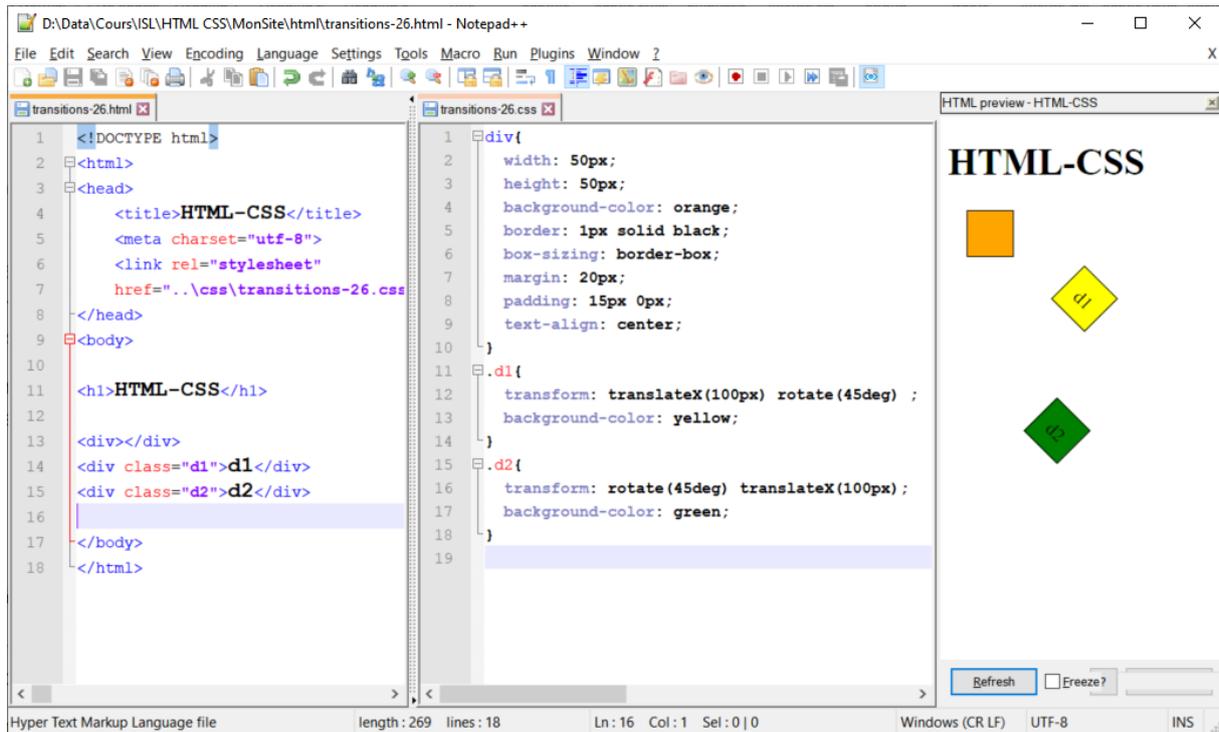
Définir une matrice de transformation avec matrix()

La fonction `matrix()` va nous permettre de définir notre propre matrice de transformation. Son utilisation est réservée aux personnes qui possèdent un bon degré de connaissance des transformations ET en mathématiques. Comme son usage est très marginal, je ne pense pas qu'il soit pertinent d'expliquer son fonctionnement en détail dans ce cours.

Appliquer plusieurs transformations d'un coup

Nous allons pouvoir définir plusieurs transformations à appliquer à un élément avec `transform`. Pour cela, il va suffire d'indiquer les différents effets de transformation à la suite.

Notez bien ici que les transformations ne vont pas toutes s'effectuer en même temps mais plutôt les unes à la suite des autres selon leur ordre de déclaration.



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>HTML-CSS</title>
5   <meta charset="utf-8">
6   <link rel="stylesheet"
7     href="..\css\transitions-26.css"
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div></div>
14 <div class="d1">d1</div>
15 <div class="d2">d2</div>
16
17 </body>
18 </html>

```

```

1 div{
2   width: 50px;
3   height: 50px;
4   background-color: orange;
5   border: 1px solid black;
6   box-sizing: border-box;
7   margin: 20px;
8   padding: 15px 0px;
9   text-align: center;
10 }
11
12 .d1{
13   transform: translateX(100px) rotate(45deg) ;
14   background-color: yellow;
15 }
16
17 .d2{
18   transform: rotate(45deg) translateX(100px);
19   background-color: green;
20 }

```

Expliquons le code ci-dessus. Ici, on applique les mêmes transformations à nos deux **div** mais pas dans le même ordre.

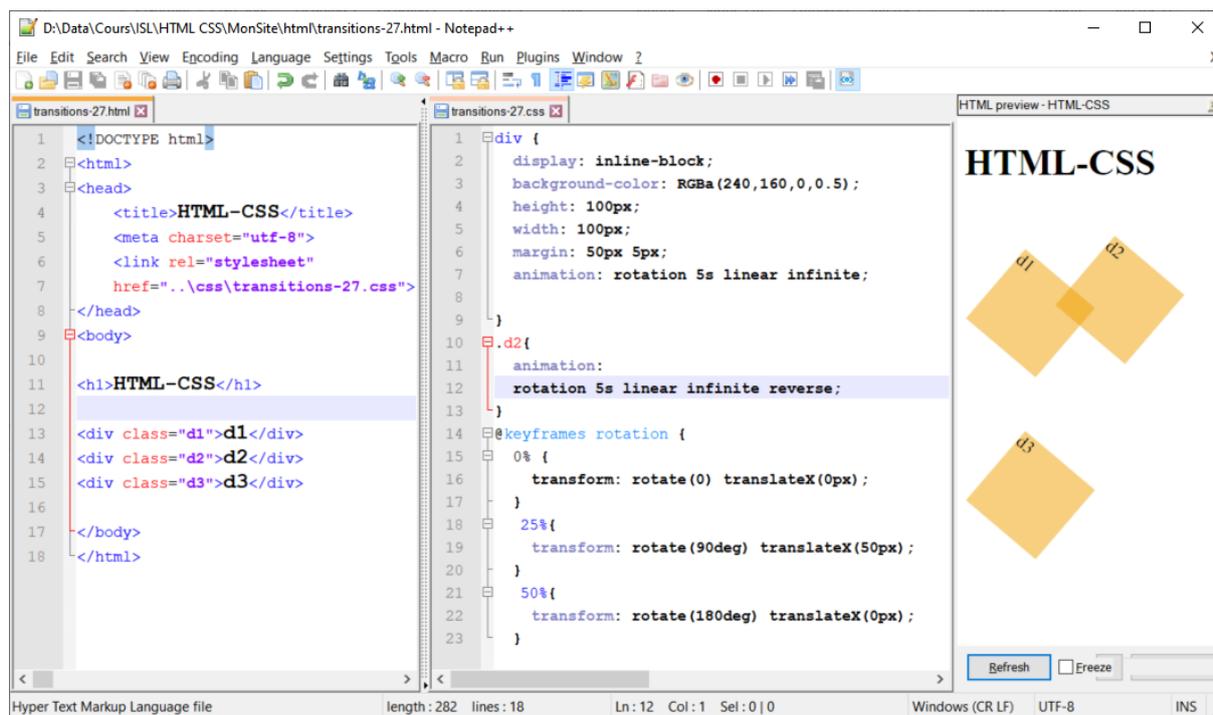
Le premier **div** auquel on applique une transformation va d'abord effectuer une translation de 100px vers la droite puis une rotation de 45 degrés dans le sens horaire.

Le deuxième **div**, au contraire, va lui d'abord effectuer une rotation de 45 degrés puis une translation de 100px. La différence est ici qu'après sa rotation l'axe horizontal du div (qu'il serait plus correct d'appeler l'axe des abscisses ou axe des X) est également incliné de 45 degrés et la translation va se faire selon cet axe.

Notre second **div** va donc être décalé de 100px selon cet axe incliné à 45 degrés.

Animer des transformations

Notez enfin qu'on va tout à fait pouvoir utiliser les transformations au sein d'animations en CSS. Cela va nous permettre d'ajouter un certain dynamisme à nos pages web.



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>HTML-CSS</title>
5   <meta charset="utf-8">
6   <link rel="stylesheet"
7     href="..\css\transitions-27.css">
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1">d1</div>
14 <div class="d2">d2</div>
15 <div class="d3">d3</div>
16
17 </body>
18 </html>
  
```

```

1 div {
2   display: inline-block;
3   background-color: RGBA(240,160,0,0.5);
4   height: 100px;
5   width: 100px;
6   margin: 50px 5px;
7   animation: rotation 5s linear infinite;
8 }
9
10 .d2{
11   animation:
12     rotation 5s linear infinite reverse;
13 }
14 @keyframes rotation {
15   0% {
16     transform: rotate(0) translateX(0px);
17   }
18   25%{
19     transform: rotate(90deg) translateX(50px);
20   }
21   50%{
22     transform: rotate(180deg) translateX(0px);
23   }
  
```

Les transformations 3D

Finalement, vous devez savoir qu'on va également pouvoir créer des transformations en 3D, c'est-à-dire en rajoutant un axe Z qui va créer une perspective. Cet axe va nous permettre de simuler une profondeur et on va pouvoir ainsi faire comme si des éléments se rapprochaient ou s'éloignaient de l'utilisateur.

Avec les transformations 3D, nous commençons à toucher à des choses vraiment complexes en CSS et qui ne sont pas forcément pertinentes dans le cadre de ce cours complet car elles sont très peu utilisées et très spécifiques.

J'aborde ce sujet par souci d'exhaustivité mais je vais me contenter de vous expliquer rapidement le principe de fonctionnement des transformations 3D et vous donner quelques exemples qui me semblent les plus pertinents car encore une fois je ne pense pas qu'il soit pertinent de s'arrêter trop longtemps sur ces notions et d'essayer à tout prix de comprendre comment elles fonctionnent en détail.

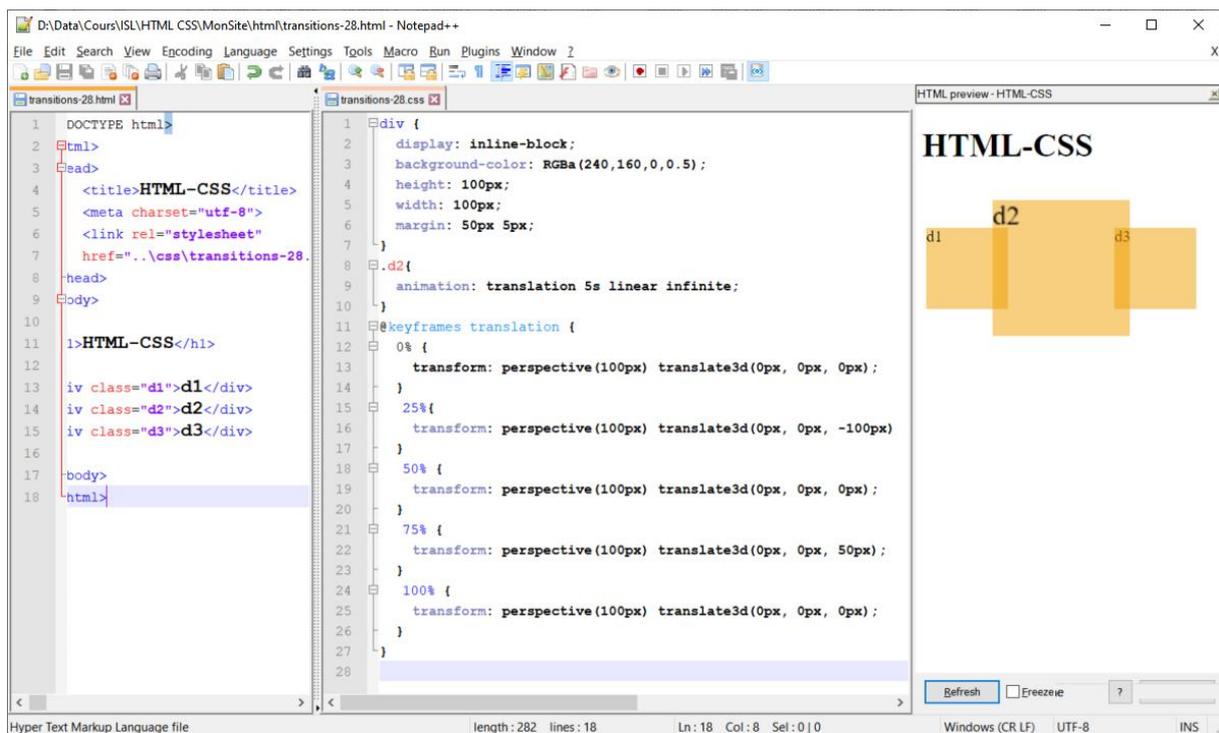
Pour qu'une transformation 3D ait l'effet visuel escompté, il va déjà falloir créer un effet de perspective c'est-à-dire créer une impression d'espace 3D. Pour cela, nous allons utiliser la fonction `perspective()` avec nos effets de transformation.

La valeur passée à la fonction `perspective()` va définir l'intensité de l'effet 3D. On peut considérer qu'elle va représenter la distance entre l'utilisateur et l'élément. Plus la valeur passée à `perspective()` va être grande, plus l'élément sera éloigné de l'utilisateur au départ et moins l'effet 3D sera visible.

Ensuite, nous allons pouvoir définir nos effets de transformation 3D : mises à l'échelle en 3D, translations 3D ou rotations 3D. Nous allons également pouvoir créer nos propres transformations 3D grâce à la fonction `matrix3d()` mais il faudra pour cela comprendre comment fonctionne le calcul matriciel en mathématiques.

Pour chaque effet de transformation cité ci-dessus, nous allons soit pouvoir utiliser les fonctions `scale3d()`, `translate3d()` et `rotate3d()`, soit utiliser les notations longues `scaleX()`, `scaleY()`, etc. et en les complétant avec `scaleZ()`, `translateZ()` et `rotateZ()`.

Voici deux exemples de translation et de rotation 3D qu'on va effectuer durant une animation :



```

1 DOCTYPE html
2 <html>
3 <head>
4 <title>HTML-CSS</title>
5 <meta charset="utf-8">
6 <link rel="stylesheet"
7 href="..\css\transitions-28.
8 </head>
9 <body>
10
11 <h1>HTML-CSS</h1>
12
13 <div class="d1">d1</div>
14 <div class="d2">d2</div>
15 <div class="d3">d3</div>
16
17 </body>
18 </html>

```

```

1 .div {
2 display: inline-block;
3 background-color: RGBA(240,160,0,0.5);
4 height: 100px;
5 width: 100px;
6 margin: 50px 5px;
7 }
8 .d2 {
9 animation: translation 5s linear infinite;
10 }
11 @keyframes translation {
12 0% {
13 transform: perspective(100px) translate3d(0px, 0px, 0px);
14 }
15 25% {
16 transform: perspective(100px) translate3d(0px, 0px, -100px);
17 }
18 50% {
19 transform: perspective(100px) translate3d(0px, 0px, 0px);
20 }
21 75% {
22 transform: perspective(100px) translate3d(0px, 0px, 50px);
23 }
24 100% {
25 transform: perspective(100px) translate3d(0px, 0px, 0px);
26 }
27 }
28

```

HTML-CSS

d1 d2 d3

Refresh Freeze ?

Hyper Text Markup Language file length: 282 lines: 18 Ln: 18 Col: 8 Sel: 0 | 0 Windows (CR LF) UTF-8 INS