Grid layout cont'd

Child Elements or Items of grid :

By default, a container has one grid item for each column in each row. However, the grid items can be styled so that they will span multiple columns and/or rows. Some very common properties we may specify are **background-color, border, padding; font-size, text-align** apart from the specified as under:

1. grid-column Property

This property specifies on which column(s) to place an item. We specify where the item will start, and where the item will end. To place an item, either refer to *line numbers*, or use the keyword "span" to define how many columns the item will span. For example:

.item {grid-column: 1 / 4;}

will make "item" to start on column 1 and end before column 4

grid-column: 1 / span 3;

will make "item" to start on column 1 and span 3 columns

grid-column: 2 / span 2;

will make "item" to start on column 2 and span 2 columns

2. grid-row Property:

This property defines on which row to place an item. We specify where the item will start, and where the item will end. To place an item, either refer to *line numbers*, or use the keyword "span" to define how many rows the item will span. For example:

.item{grid-row: 1 / 4;}

will make "item" to start on row 1 and end before row 4

grid-row: 1 / span 3;

will make "item" to start on row 1 and span 3 rows

grid-row: 2 / span 2;

will make "item" to start on row 2 and span 2 rows

Example:

```
<html>
<head>
<style>
.grid-container {
  display: grid;
  grid-gap: 10px;
  background-color: red;
  padding: 10px;
}

.grid-item {
  background-color: green;
  text-align: center;
  padding: 20px;
  font-size: 30px;
}

.item3 {
  grid-column: 1/ span 2;
  grid-row: 2;
}

.item2 {
  grid-column: 3;
  grid-row: 1 / span 2;
}

.item5 {
  grid-column: 1 / span 3;
  grid-row: 3;
}
</style>
</head>
<body>

<h1>Five Items Grid Layout with varying row and column span</h1>
<div class="grid-container">
  <div class="grid-item item1">1</div>
  <div class="grid-item item2">2</div>
  <div class="grid-item item3">3</div>
  <div class="grid-item item4">4</div>
  <div class="grid-item item5">5</div>
</div>
<p>Direct child elements(s) of the grid container automatically becomes grid items.</p>
<p>Item 2, 3, 5 have set to span multiple columns or rows.</p>
</body>
</html>
</html>
```

3. grid-area Property

This property may be used as a shorthand property for the *grid-row-start, grid-column-start, grid-row-end* and the *grid-column-end* properties. For example:

.item { grid-area: 1 / 2 / 5 / 6; }

Will make "item" to start on row-line 1 and column-line 2, and end on row-line 5 and column line 6

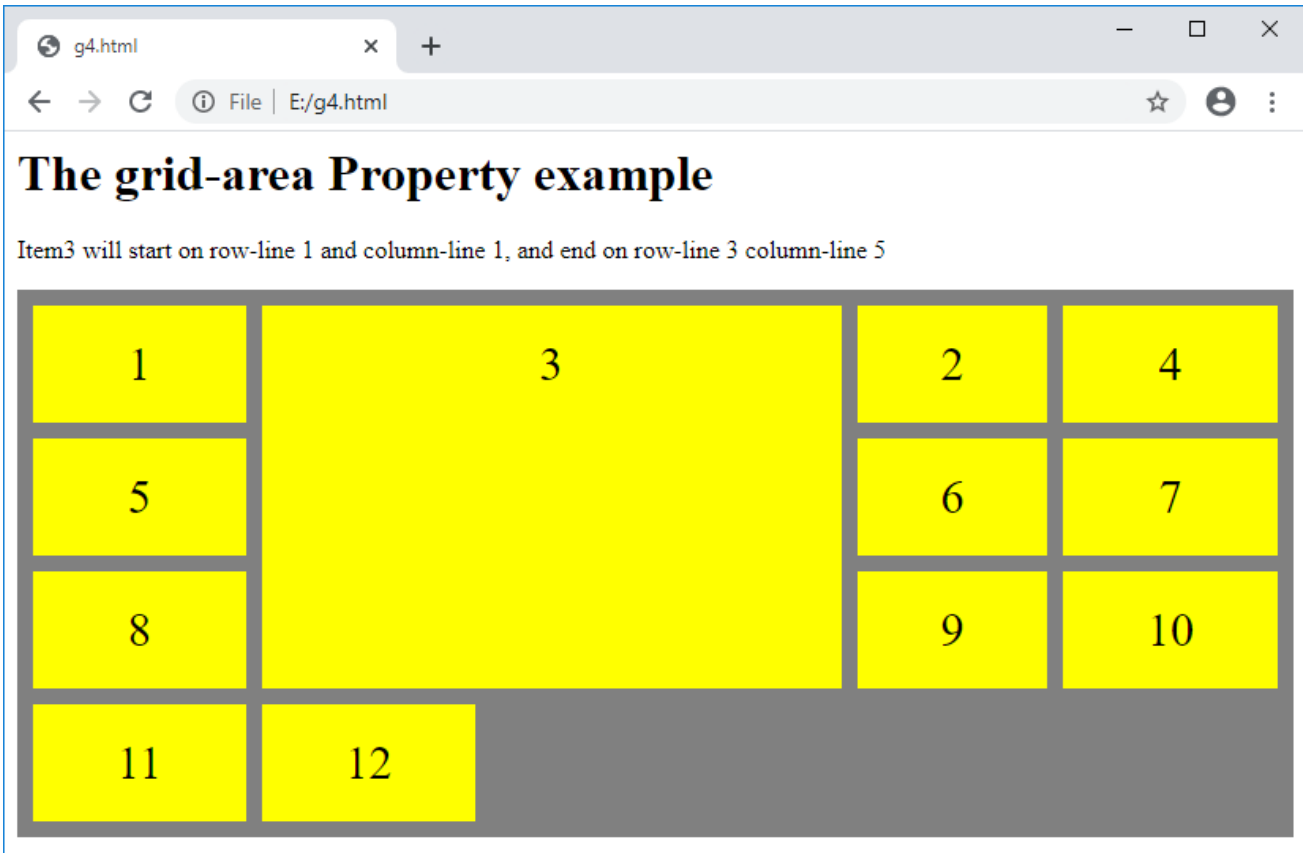
grid-area: 2 / 1 / span 2 / span 3;

Will Make "item" to start on row-line 2 and column-line 1, and span 2 rows and 3 columns

Example: Grid Area Property

```
<html>
<head>
<style>
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto auto auto auto;
  grid-gap: 10px;
  background-color: gray;
  padding: 10px;
}

.grid-container > div {
  background-color: yellow;
  text-align: center;
  padding: 20px 0;
  font-size: 30px;
}

.item3 {
  grid-area: 1 / 2 / 4 / 5;
}
</style>
</head>
<body>

<h1>The grid-area Property example</h1>

<p>Item3 will start on row-line 1 and column-line 1, and end on row-line 3 column-line 5</p>

<div class="grid-container">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
  <div class="item4">4</div>
  <div class="item5">5</div>
  <div class="item6">6</div>
  <div class="item7">7</div>
  <div class="item8">8</div>
  <div class="item9">9</div>
```

```
  <div class="item10">10</div>
   <div class="item11">11</div>
   <div class="item12">12</div>
</div>
</body>
</html>
```



Naming Grid Items

The **grid-area** property is used to assign names to grid items.

Further, Named grid items can be referred to by the **grid-template-areas** property of the grid container.

Each row is defined by **apostrophes** (' ') and the columns in each row is defined inside the apostrophes, separated by a **space**. i.e. To define two rows, define the column of the second row inside another set of apostrophes.

A **period** (.) sign represents a grid item with no name.

Example-1 :

```
.item3 {
  grid-area: myArea;
}
.grid-container {
  grid-template-areas: 'myArea myArea myArea myArea';
}
```

Here, item3 gets the name "myArea" and spans all four columns in a four columns grid layout.

Example-2 :

```
.item3 {
  grid-area: myArea;
}
.grid-container {
  grid-template-areas: 'myArea myArea myArea ..';
}
```

Here, "myArea" span three columns in a five columns grid layout as period signs represent items with no name

Example-3 :

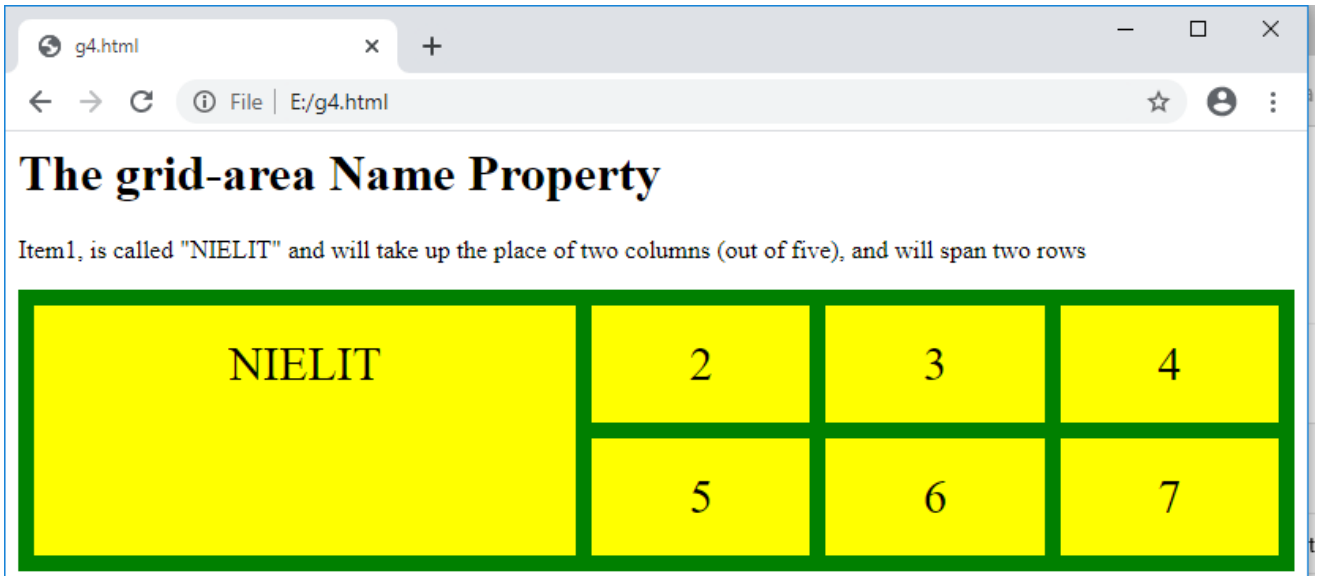
```
.grid-container {
  grid-template-areas: 'myArea myArea . . .' 'myArea myArea . . .';
}
```

Will Make "item1" to span two columns and two rows

Example:

```
<html>
<head>
<style>
.item1 {
  grid-area: NIELIT;
}
.grid-container {
  display: grid;
  grid-template-areas: 'NIELIT NIELIT . . .' 'NIELIT NIELIT . . .';
  grid-gap: 10px;
  background-color: green;
  padding: 10px;
}
.grid-container > div {
  background-color: yellow;
  text-align: center;
  padding: 20px 0;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The grid-area Name Property</h1>
<p>Item1, is called "NIELIT" and will take up the place of two columns (out of five), and will span two rows</p>
<div class="grid-container">
  <div class="item1">NIELIT</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
  <div class="item4">4</div>
  <div class="item5">5</div>
```

```
  <div class="item6">6</div>
  <div class="item7">7</div>
</div>
</body>
</html>
```



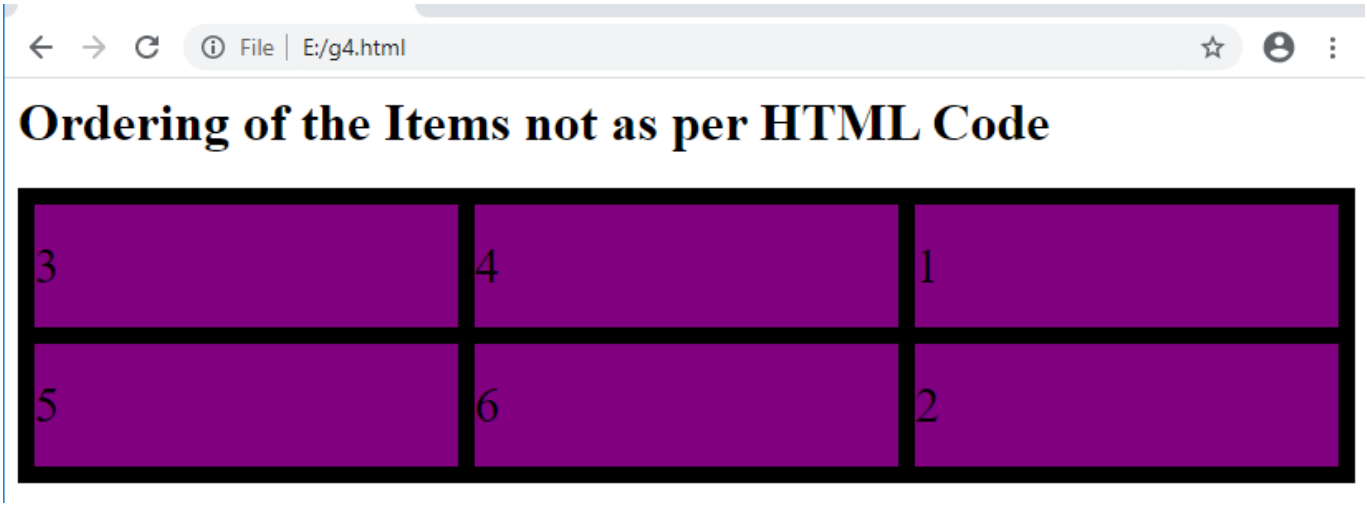
Order of the Items

The Grid Layout allows us flexibility to position the items anywhere we like. The first item in the HTML code does not have to be appearing as the first item in the grid.

Example

```
<html>
<head>
<style>
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto;
  grid-gap: 10px;
  background-color: black;
  padding: 10px;
}

.grid-container > div {
  background-color: purple;
  text-align: left;
  padding: 20px 0;
  font-size: 30px;
}

.item1 { grid-area: 1 / 3 / 2 / 4; }
.item2 { grid-area: 2 / 3 / 3 / 4; }
.item3 { grid-area: 1 / 1 / 2 / 2; }
.item4 { grid-area: 1 / 2 / 2 / 3; }
.item5 { grid-area: 2 / 1 / 3 / 2; }
.item6 { grid-area: 2 / 2 / 3 / 3; }
```

```
</style>
</head>
<body>

<h1>Ordering of the Items not as per HTML Code</h1>
<div class="grid-container">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
  <div class="item4">4</div>
  <div class="item5">5</div>
  <div class="item6">6</div>
</div>
</body>
</html>
```



Assignment

1.What are different properties of child element of grid?
2.What is naming properties of GRID items?