CSS Grid and Flexbox

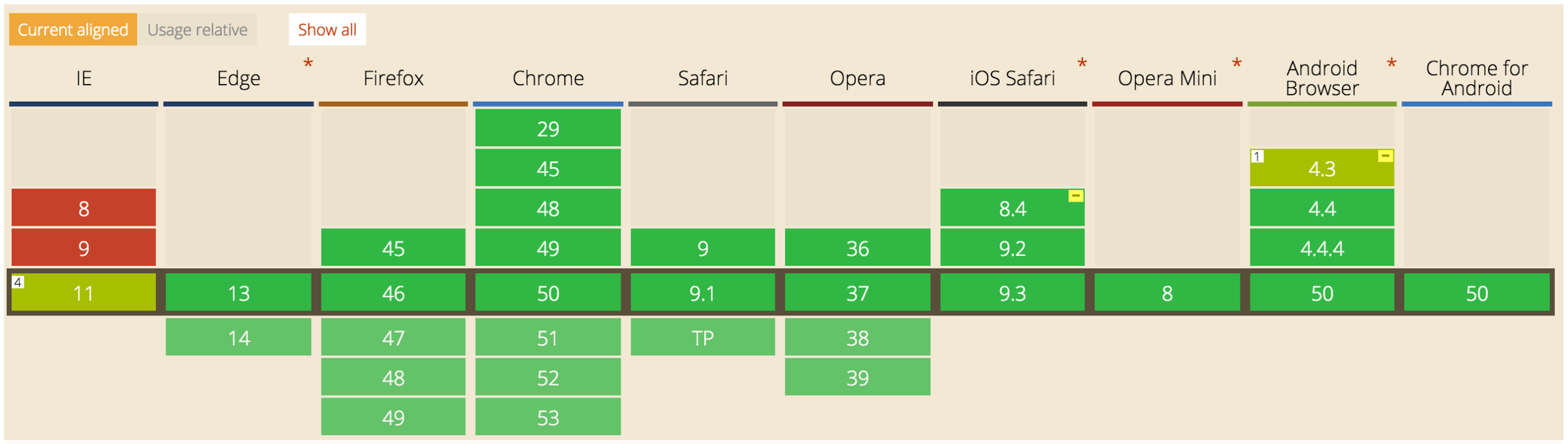
Positioning Elements

- CSS0: Tables
- CSS2: Floats
- CSS3: Flexbox

(Flexible Box) module (currently a W3C Last Call Working Draft) aims at providing a more efficient way to lay out, align and distribute space among items in a container, even when their size is unknown and/or dynamic (thus the word "flex")

Positioning Elements

Flexbox support is very good with the exception of Internet Explorer (surprise)



Positioning Elements

All flexbox containers start with this property

```
.container {
  display: flex;
}
```

```
<div class="container">
  <div class="item"></div>
</div>
```

Positioning Elements

Children of flexbox containers

.item {

flex-grow: <number>

}

Controls the ability for the child to grow in the container.

Example 1

Positioning Elements

Children of flexbox containers

.item {

flex-shrink: <number>

}

Controls the ability for the child to shrink in the container.

Example 2

Positioning Elements

Children of flexbox containers

.item {

align-self: flex-start | flex-end | center | baseline | stretch

}

Controls where the item is displayed in the container.

Example 3

Positioning Elements

Flexbox containers

.container {

flex-direction: row | row-reverse | column | column-reverse

}

Controls the order of items in the container.

Example 4

Positioning Elements

Flexbox containers

.container {

flex-wrap: nowrap | wrap | wrap-reverse

}

.item {

width: <%>

}

Controls how items wrap in the container.

Example 5

Positioning Elements

Flexbox containers

.container {

justify-content: flex-start | flex-end | center | space-between | space-around

}

.item {

width: <px | %>

}

Controls spacing on items wrap in the container.

Example 6

Positioning Elements

Flexbox containers

.container {

align-items: flex-start | flex-end | center | baseline | stretch

}

Controls vertical placement of items in the container.

Example 7

Positioning Elements

Flexbox containers

.container {

align-content: flex-start | flex-end | center | space-between | space-around | stretch

}

.item{

width: <px | %>

}

Controls placement of items in the container.

Example 8

Positioning Elements

Grid containers

Two-dimensional grid-based layout system.

Working draft status. Not production ready.

Can be enabled with browser config options.

Positioning Elements

Grid containers

.container{

display: grid;

grid-template-columns: [line-name] <px | % | auto> | ...

grid-template-rows: [line-name] <px | % | auto> | ...

}

Columns can have more than one name.

Positioning Elements

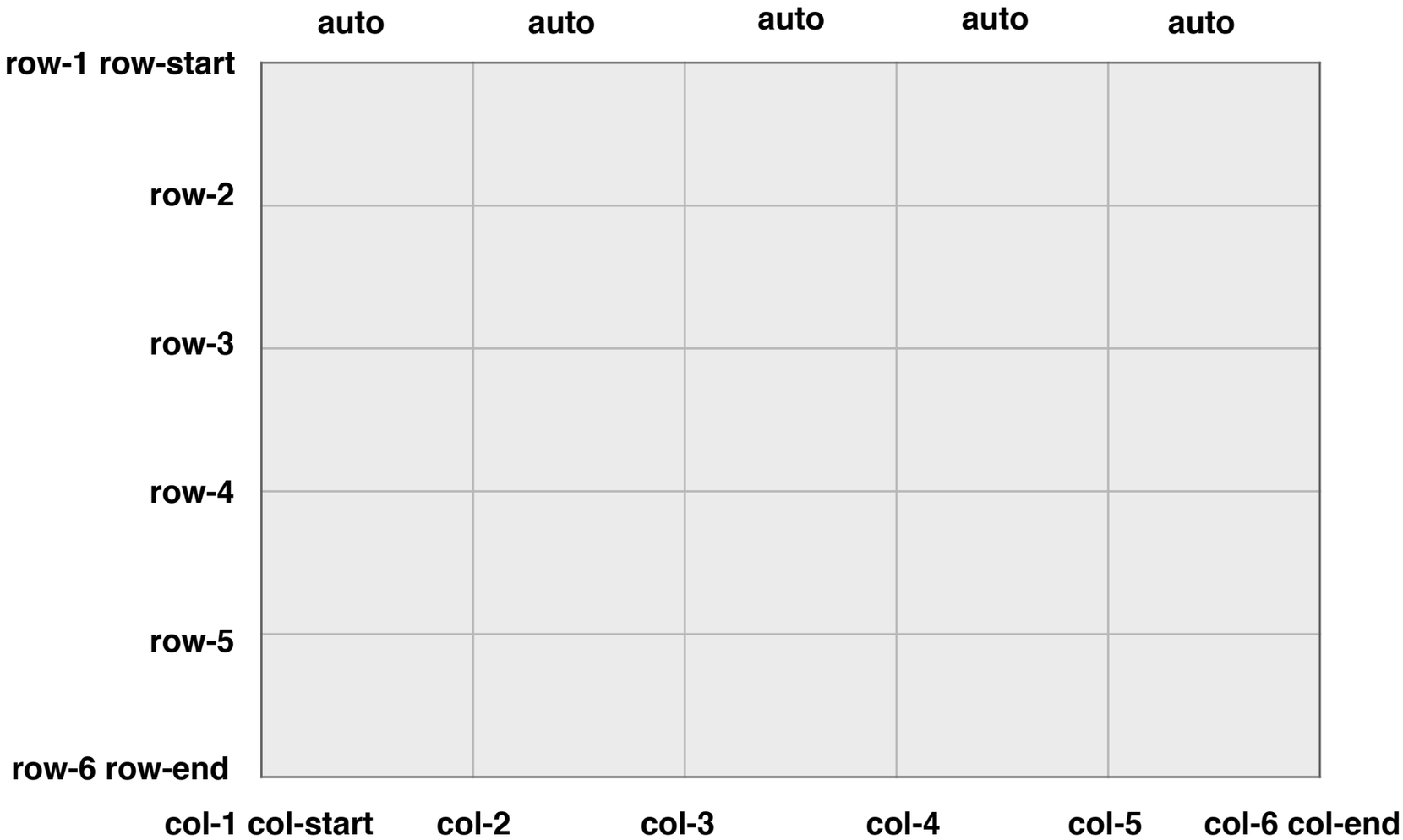
Grid containers

grid-template-columns: [col-1 col-start] auto [col-2] auto [col-3] auto [col-4] auto [col-5] auto [col-6 col-end];

grid-template-rows: [row-1 row-start] auto [row-2] auto [row-3] auto [row-4] auto [row-5] auto [row-6 row-end];

Positioning Elements

Grid containers



Positioning Elements

Grid containers

.item{

grid-column-start: number | name | auto

grid-column-end: ...

grid-row-start: ...

grid-row-end: ...

}

Tell the grid element where to position itself.

Example 9

Positioning Elements

Grid containers

.container{

grid-template-areas: "string string ..."

}

Configure areas of a grid. Items can be positioned on a grid area.

Positioning Elements

Grid containers

grid-template-areas: "header header header header header header"

"left-side main main main right-side"

"left-side main main main right-side"

"left-side main main main right-side"

"footer footer footer footer footer footer";

Positioning Elements

Grid containers

.item{

grid-area: name

}

Tell the grid element where to position itself.

Example 10

Resources

CSS3 Demo

https://github.com/metric152/css-demo

Guides

https://css-tricks.com/snippets/css/complete-guide-grid/

https://css-tricks.com/snippets/css/a-guide-to-flexbox/