**Модуль 7**. **Каскадные таблицы стилей. Размеры элемента**

Ширина и высота блочных элементов.

Свойства CSS **width** и **height** - устанавливают ширину и высоту блочных элементов. Ширина и высота элемента может быть задана следующими способами:

* auto - Размеры элемента определяется его содержанием. (по умолчанию)
* % - Размеры элемента задаётся в процентах от высоты/ширины элемента родителя.
* px - Размеры элемента задаётся в пикселях или любых других единицах измерения принятых в CSS.

Пример в файле width\_height.html

Управление содержанием элемента.

Что делать с содержанием элемента, если оно превышает его размер?

Если элементу присвоены точные значения высоты и ширины (height, width) а его содержание, например длинный текст, не вмещается в указанных пределах, то по умолчанию такой элемент растягивается до нужных размеров, что не всегда на руку веб-мастеру. На помощь приходит атрибут overflow, который указывает браузеру, что делать с элементом в таких случаях.

Свойство overflow может иметь следующие значения:

* visible - Элемент растягивается до необходимых размеров. (по умолчанию)
* hidden - Содержание элемента "обрезается" видна лишь та его часть что помещается в элементе.
* scroll - Добавляются полосы прокрутки (всегда! даже если содержание помещается в пределах элемента).
* auto - Полосы прокрутки добавляются при необходимости.

Пример в файле visible.html

Курсоры.

Свойство CSS **cursor** позволяет установить нестандартный вид курсора для того или иного элемента - блока, текста, рисунка.. то есть когда пользователь наведет курсор на такой элемент он, курсор то бишь, поменяет свой вид.

Курсоры можно выбирать как стандартные, так и подгружать свои собственные пользовательские курсоры - файлы в формате cur, ani или svg.

Стандартные курсоры.

Вы наверняка обратили внимание на то, что по умолчанию, на странице, курсор приобретает вид в зависимости от специфики элемента, на который он нацелен, например, если навести курсор на ссылку он отображается в виде указателя обычно "руки", если на некий текст, то курсор становится вида "работа с текстом", а если просто елозить по пустому месту на странице, то курсор имеет обыкновенный вид "стрелка".

Так вот, для того чтобы изменить внешний вид курсора свойству **cursor** необходимо указать одно из возможных значений:

* auto - курсор назначается автоматически в зависимости от специфики элемента.(по умолчанию)
* crosshair - перекрестие
* default - встроенный(основной)
* e-resize - стрелка на "восток"
* hand - указатель аналогия pointer
* help - помощь
* move - перемещение
* n-resize - переместить на "север"
* ne-resize - переместить на "северо-восток"
* nw-resize - переместить на "северо-запад"
* pointer - указатель
* progress - продолжение операции
* s-resize - переместить на "юг"
* se-resize - переместить на "юго-восток"
* sw-resize - переместить на "юго-запад"
* text - текст
* w-resize - переместить на "запад"
* wait – ожидание

Пользовательские курсоры.

Для того чтобы курсор приобрёл нестандартный вид его необходимо подгрузить присвоив свойству **cursor** значение: **url("путь к курсору").**

Например:

div { cursor : url("my.cur"); }

Так же можно указать несколько пользовательских курсоров, через запятую, в этом случае браузер будет пытаться отобразить первый из перечисленных, если у него это не получится, возьмется за второй.. и т. д. А не получится может из-за того, что браузер не поддерживает определённый формат файла-курсора. Как уже говорилось выше можно использовать файлы в формате cur, ani или svg - такие курсоры поддерживаются, начиная с IE6, Firefox 1.5. Однако Firefox 1.5 не поддерживает формат ani, а IE6 в свою очередь, не понимает формат svg.

Поэтому список курсоров составляют из файлов разных форматов.. например, так:

* + div { cursor : url("my.cur"), url("my.svg"); }

И уж что б совсем обезопасить себя в конце списка рекомендуется ставить один из курсоров из стандартного набора.

Вот так:

div { cursor : url("my.cur"), url("my.svg"), help; }

Показ элементов.

Свойство **display** указывает браузеру, как тот или иной элемент должен быть показан на странице, другими словами определяет параметры вывода браузером элемента на экран.

**Display** имеет множество возможных значений, однако, большинство из них поддерживаются далеко не всеми браузерами, поэтому мы поговорим лишь о тех которые будут корректно работать во всех браузерах, а таковых всего три:

* block - Показывает элемент как блочный. Происходит перенос строк вначале и в конце элемента.
* inline - Показывает элемент как встроеный. Элемент не переносятся на следующую строку.
* inline-block - показывает элемент как блочный «внутри» и как встроенный «снаружи»
* none - Запрещает показ элемента. Элемент "удаляется" из общего потока, его занимаемое пространство на экране не резервируется.

Значения **block**, **inline** и **inline-block** свойства **display** позволяет самостоятельно указывать, какие элементы мы хотим сделать строчными, а какие блочными, что позволяет решить ряд задач при верстке сайта с помощью CSS.

Пример в файле display.html

Видимость элемента.

Свойство CSS **visibility** предназначено для отображения или скрытия элемента, включая рамку вокруг него и фон. При скрытии элемента с помощью свойства **visibility** элемент становится, не виден, однако место, которое он занимает, остается за ним.

Возможные значения свойства visibility:

* inherit - Элемент отображается так как указанно в элементе родителе.
* visible - Отображает элемент. (по умолчанию).
* hidden - Скрывает элемент. В отличии от display: none оставляет за элементом площадь которую он занимает. Проще говоря если display: none "удаляет" элемент то hidden делает его "прозрачным".
* collapse - Скрывает колонки и строки таблицы идентично display: none, Если свойство применяется к другим элементам, не имеющим отношения к таблицам, то результат будет таким же, как hidden.

Пример в файле visibility.html

Обтекание элемента

Свойство **float** определяет, по какой стороне будет выравниваться - всплывать элемент, при этом остальные элементы будут обтекать его с других сторон.

Выравнивание элемента происходит по краям родительского элемента или же по краям окна браузера.

Значения свойства **float**:

* left - Выравнивает элемент по левому краю, остальные элементы обтекают его справа.
* right - Выравнивает элемент по правому краю, остальные элементы обтекают его слева.
* none - Выравнивание элемента не задается. (по умолчанию)

Пример в файле float.html

Запрет обтекания

Свойство **clear** запрещает обтекание элемента с левой и/или правой стороны. Если для элемента установлено обтекание с помощью свойства **float,** то **clear** отменяет обтекание данного элемента с указанных сторон.

Значения:

* both - Запрещает обтекание элемента с левой и правой стороны.
* left - Запрещает обтекание элемента с левой стороны.
* right - Запрещает обтекание элемента с правой стороны.
* none - Запрет на обтекание элемента не задается.(по умолчанию)

Пример в файле clear.html