Лекция: Flexible Box, Grid Layout, Анимация в CSS через keyframes

# Введение

Сегодня мы рассмотрим три важные темы в CSS: Flexible Box (Flexbox), Grid Layout и Анимацию через keyframes. Эти инструменты позволяют создавать адаптивные, динамичные и визуально привлекательные веб-страницы.

# Flexible Box (Flexbox)

Flexbox — это макетная модель, которая позволяет автоматически распределять пространство между элементами контейнера и управлять их выравниванием, независимо от их размера. Он идеально подходит для создания адаптивных и гибких макетов.

## Основные свойства Flexbox

1. display: flex — превращает элемент в flex-контейнер.  
2. flex-direction — задает направление оси (row, column, row-reverse, column-reverse).  
3. justify-content — выравнивание элементов вдоль основной оси (flex-start, center, flex-end, space-between, space-around, space-evenly).  
4. align-items — выравнивание по поперечной оси (flex-start, center, flex-end, baseline, stretch).  
5. flex-wrap — позволяет элементам переноситься на новую строку при переполнении (nowrap, wrap, wrap-reverse).

# Grid Layout

CSS Grid Layout — это мощный инструмент для создания двухмерных сеточных макетов. В отличие от Flexbox, который работает с одномерными макетами (строка или колонка), Grid позволяет управлять как строками, так и колонками одновременно.

## Основные свойства Grid Layout

1. display: grid — превращает элемент в grid-контейнер.  
2. grid-template-columns / grid-template-rows — определяют количество и размер колонок и строк.  
3. grid-gap — задает расстояние между ячейками сетки.  
4. justify-items / align-items — выравнивание элементов внутри ячеек (start, center, end, stretch).  
5. grid-auto-flow — управляет автоматическим размещением элементов (row, column, dense).

# Анимация в CSS через keyframes

Анимации в CSS позволяют плавно изменять свойства элементов с течением времени. Для создания анимации используется правило @keyframes, где задаются шаги изменения свойств.

## Шаги создания анимации

1. Определить анимацию с помощью @keyframes, описав ключевые кадры (from, to или проценты от 0% до 100%).  
2. Привязать анимацию к элементу с помощью свойства animation.

### Пример использования keyframes

@keyframes example {  
 from {opacity: 0;}  
 to {opacity: 1;}  
}

В этом примере анимация будет постепенно менять прозрачность элемента с 0 до 1.

## Атрибуты свойства animation

1. animation-name — имя анимации (example).  
2. animation-duration — длительность анимации (например, 2s).  
3. animation-timing-function — функция распределения времени (linear, ease, ease-in, ease-out, ease-in-out).  
4. animation-delay — задержка перед началом анимации.  
5. animation-iteration-count — количество повторений анимации (infinite, 1, 2 и т.д.).  
6. animation-direction — направление анимации (normal, reverse, alternate, alternate-reverse).  
7. animation-fill-mode — сохраняет конечное состояние анимации (none, forwards, backwards, both).  
8. animation-play-state — управляет воспроизведением (paused, running).

### Пример использования animation

div {  
 animation-name: example;  
 animation-duration: 2s;  
 animation-timing-function: ease-in;  
 animation-iteration-count: infinite;  
}

# Заключение

Сегодня мы изучили основы работы с Flexbox и Grid Layout, а также узнали, как создавать анимации с использованием ключевых кадров в CSS. Эти инструменты позволяют делать страницы более гибкими, адаптивными и интерактивными.