**1.1. Зачем вебу понадобился CSS?**

**HTML** — язык структуры, но не стиля. Сайты 90-х были “голыми”:

* Только стандартный шрифт и цвет,
* Нет “шапки”, подвала, колонок, фирменного оформления,
* Даже заголовки выглядели почти как обычный текст.

**Проблема:** разработчики хотели управлять внешним видом, но единственный способ — жёстко прописывать атрибуты прямо в теги.  
Пример:

html

КопироватьРедактировать

<font color="red" size="5">Важная новость!</font>

* **Минусы:**  
  — “Вечный” копипаст кода,  
  — Оформление смешано со структурой,  
  — Любое изменение цвета/размера — редактирование по всему сайту вручную.

**Таблицы в HTML:**  
Вместо layout-систем — “скелет” сайта на <table>, <tr>, <td>, для любого макета, не только данных.

html

КопироватьРедактировать

<table>

<tr>

<td>Меню</td>

<td>Контент</td>

</tr>

</table>

* **Проблема:** сложно изменять, тяжело адаптировать, не “семантично”.

**1.2. Рождение CSS: ключевой момент для Интернета**

**1994–1996:**

* Ведущие умы W3C (главный комитет веба) видят хаос в “раскрашивании” сайтов и понимают: оформление должно быть *отдельно* от структуры.
* Хокон Виум Ли (Håkon Wium Lie) предлагает **CSS** — таблицы стилей, которые можно менять в одном месте, а применяются к любым html-страницам.

**Первый стандарт CSS (CSS1, 1996):**

* Цвета текста и фона,
* Шрифты и их размеры,
* Простые отступы (margin, padding),
* Рамки, простая типографика.

**Преимущество:**

* Можно менять оформление всех заголовков или абзацев на сайте одним-двумя правилами, а не редактировать каждую страницу отдельно.

**1.3. “Дикие 90-е”: хаки, войны браузеров и первые стили**

**Основная проблема:**  
— Браузеры реализовывали CSS по-разному, часто с багами.  
— В Internet Explorer, Netscape Navigator, Opera стили работали по-разному!

**Появились “хаки”:**

* Особые конструкции, которые понимал только один браузер.
* “Conditional comments” — специальные комментарии для IE.

**Первые “CSS-фреймы”:**

* Дизайнеры “крутили” макеты, впервые игрались с отступами и цветами, создавали “шапки” и “подвалы”, начинали мыслить блоками.

**1.4. CSS2: рождение позиционирования и спецэффектов (1998–2000)**

* **Появилось абсолютное, относительное, фиксированное позиционирование:** теперь не только таблицы, но и “свободные” блоки!
* **Псевдоклассы и псевдоэлементы:** эффекты при наведении, оформление первой строки/буквы, стилизация форм и ссылок.
* **Стиль для печати:** возможность делать отдельный CSS для печати (только контент, убираем всё лишнее).

**Феномен:** появились первые “фирменные” сайты с уникальным стилем, стильными кнопками, вертикальными меню, спецэффектами.

**1.5. Эпоха хаоса: IE6, хаки, несовместимости**

**IE6** — символ эпохи проблем:

* CSS работал нестабильно: боковые отступы “ломались”, шрифты “убегали”, шапки съезжали.
* Стилизация скроллбаров, форм — почти невозможна.
* PNG-прозрачность отсутствовала.

**Борьба за кроссбраузерность:**

* Верстальщики вырабатывали целые “боевые уставы”, чтобы сделать сайт похожим в IE, Firefox и Opera.
* На каждом проекте был “табель хаков” — таблица для исправления бага в каждом браузере.

**1.6. CSS3: рождение современных сайтов**

**2009–2015:**

* CSS3 развивается как набор модулей: теперь можно внедрять новые свойства по частям, не дожидаясь “новой версии CSS”.
* **Что появилось:**
  + Градиенты и тени без картинок,
  + border-radius, opacity, box-shadow,
  + Переходы и анимации (transition, @keyframes),
  + Web Fonts (подключаем любой шрифт через интернет, Google Fonts),
  + Медиа-запросы (адаптивность),
  + Flexbox — новый способ компоновки,
  + :nth-child, :not, сложные селекторы,
  + Маски, фильтры, миксины (SCSS), кастомные свойства (переменные).

**Прорыв:**

* Вёрстка становится “живой” — плавные переходы, сложные карточки, анимация, мобильные сайты.
* Фреймворки (Bootstrap, Foundation, Materialize) позволяют разрабатывать быстрее и стандартизировать макеты.

**1.7. Современный CSS: эпоха гибкости, адаптивности и динамики**

**2020+:**

* Темные/светлые темы внедряются одной строкой (media query),
* CSS Variables позволяют менять всю цветовую схему в реальном времени,
* Гриды и флексы дают макетам свободу без “таблиц”,
* Анимация — не только для украшения, а как элемент UX,
* CSS-in-JS и CSS-модули делают стили локальными для компонентов,
* Поддержка accessibility, high-contrast mode,
* Гигантские сайты (Avito, VK, Google) пишутся на “слоёных” CSS-архитектурах: BEM, Atomic, ITCSS, Tailwind.

**1.8. Влияние CSS на индустрию и профессию**

* Рождение профессии “верстальщик”, переход к “front-end-разработчику”.
* Дизайнер теперь может “программировать интерфейс”, а не только рисовать.
* Быстрее запуск проектов — темы, шаблоны, копипаст, обмен стилистикой между сайтами.
* Веб стал “живым”, быстрым, разнообразным и гибким.

**1.9. Самые распространённые ошибки и антипаттерны по версии старых и новых верстальщиков**

* “Всё через таблицы” — до сих пор встречается, но больше не нужно!
* Жёсткая привязка цветов, размеров, шрифтов (без переменных, без адаптива).
* Использование inline-стилей вместо классов.
* Копипаст старых “IE6-хаков” в современные проекты (не работает).
* Отсутствие семантики: когда классы называются “.style1, .style2, .main2” и никто не понимает, за что они отвечают.

**1.10. Современные best practices (2024)**

* Использовать только внешние CSS-файлы и минимизировать инлайн-стили.
* Всегда писать адаптив — даже если проект маленький.
* Для шрифтов — Google Fonts или локальные .woff2 (не ttf/eot!).
* Использовать переменные для цветов, размеров, отступов.
* Проверять цвета и контрастность (accessibility!).
* Не использовать !important без причины.
* Строить архитектуру классов по методологиям (BEM, Atomic).
* Постоянно оптимизировать, минифицировать, валидировать CSS.

**1.11. Лайфхаки, советы, must-know**

* Для теста цвета — Coolors.co.
* Для анимации — cubic-bezier.com, easings.net.
* Для кроссбраузерности — CanIUse.com.
* Для CSS-миксины, функции, оптимизация — PostCSS, SCSS/SASS.
* Всегда держать DevTools под рукой: стили меняй “на лету”.

**1.12. Вопросы для самопроверки**

1. Чем отличался HTML без CSS и современный сайт?
2. Почему таблицы использовались для верстки в 90-х, и почему это плохо?
3. Что дало появление CSS3?
4. Назови 3 современных фишки CSS, которые ты используешь ежедневно.
5. Почему CSS считают “живым” языком и как он повлиял на профессию front-end?