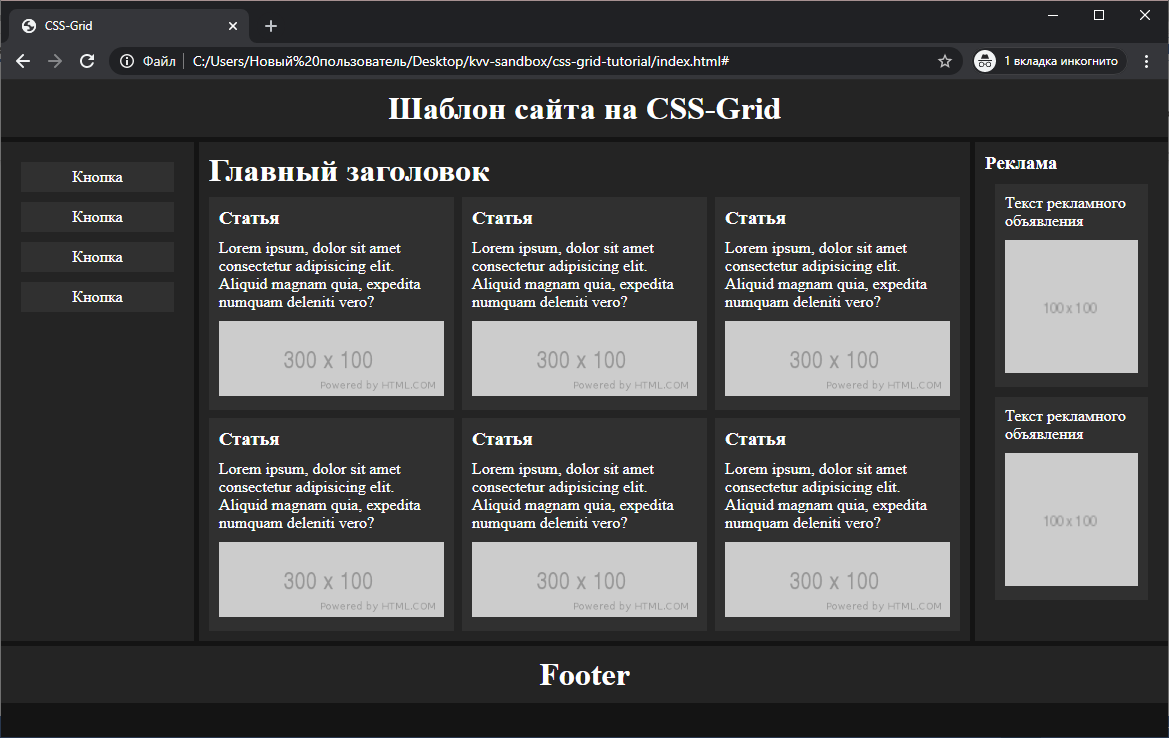
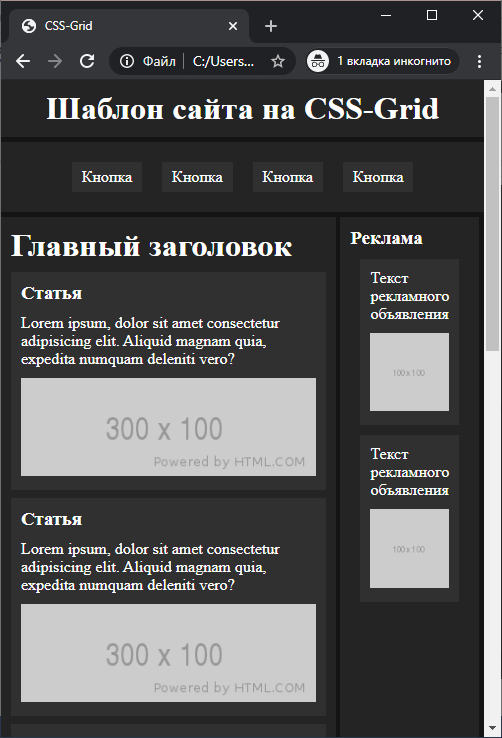
**Адаптивная верстка при помощи CSS-Grid и медиа-запросов**

В данном уроке мы сделаем адаптивный шаблон страницы, как на скриншотах:





**CSS-Grid**

CSS Grid Layout – двумерная система компоновки основанная на сетке, цель которой заключается в том чтобы полностью изменить способ проектирования пользовательских интерфейсов. Сначала мы использовали таблицы, потом обтекания (floats), позиционирование и inline-block, но все эти методы по существу являются просто хаками и опускают много важных функциональных возможностей (например, вертикальное выравнивание). Flexbox помог, но он предназначен для более простых одномерных макетов, а не для сложных двумерных (на самом деле Flexbox и Grid очень хорошо работают вместе).

**Создание макета:**

Инициализируем пустой шаблон index.html. Подключаем к шаблону стили.



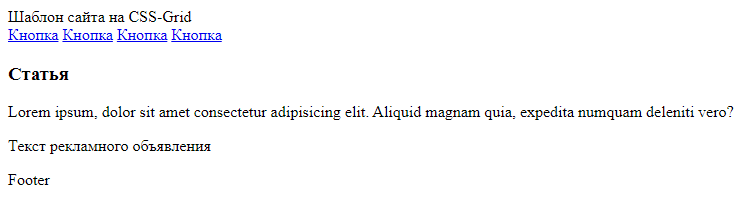
Создаем блок с классом grid-container, который будет служить родительским элементом для нашей сетки.

В grid-container помещаем блоки нужные для нашей страницы: header, menu, content, ads, footer, как показано на скриншоте.



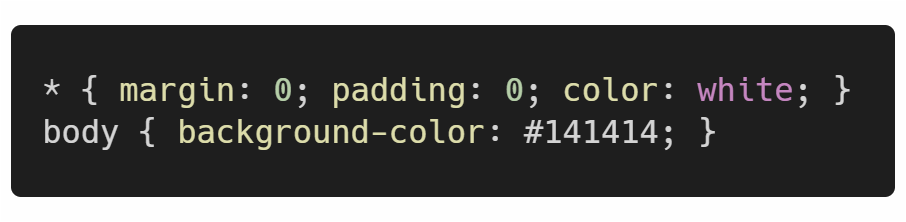
Наполним наши блоки контентом. В header необходимо добавить название страницы, в menu заглушки для возможных кнопок с классом menu-btn, в content заготовку статьи с классом article, заголовком и содержанием, а в ads шаблон рекламного объявления c классом ad.





На данный момент наша страница выглядит не так, как хотелось бы. Переходим к css-стилям.

Зададим странице темный фон, как в примере, уберем отступы, что добавляет браузер по умолчанию и зададим белый цвет шрифта.



Подключим к grid-container свойство display: grid. Именно оно заставляет его и дочерние элементы подчиняться правилам css-grid.

Свойство grid-template-columns говорит сколько колонок будут занимать блоки и какой ширины будут эти колонки. Можно указывать в процентах, пикселях или любых других единицах измерения, но правильно в css-grid указывать в фракциях (fr).

Правило grid-template-columns: 1fr 4fr 1fr означает что в нашей сетке будет три колонки суммарно в 6 фракций. Первая будет занимать одну, вторая четыре и третья одну. Все последующие блоки будут переноситься в следующий ряд и подчиняться тем же правилам.

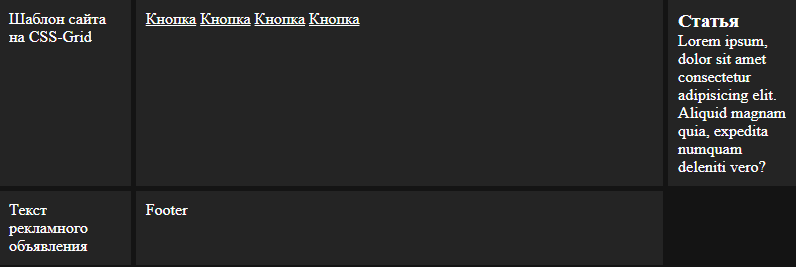
Правило grid-auto-rows задает высоту строк нашей сетки в значение max-content, что не дает блокам быть выше, чем их содержимое.

Grid-gap это способ задавать отступы между блоками в css-grid.

Также зададим небольшие отступы и более светлый цвет фона, для всех дочерних элементов.



Вот, что у нас получилось:



Не совсем то, что нам нужно. Необходимо, чтоб header и footer занимали всю ширину нашей сетки. Установим им правило grid-column: 1/4. Оно означает, что блок будет шириной от первой грани сетки, до четвертой. Так как у нас всего три блока, то у нашей сетки 4 границы столбцов.

|  |  |  |
| --- | --- | --- |
|  |  |  |

Сделаем текст footer и header жирным, выровняем по центру и увеличим размер шрифта. Вот, что у нас получилось:

