

## Создание генератора пароля с помощью html, css и js.

Генерация случайных надежных паролей — важная задача для защиты аккаунтов и конфиденциальной информации. Реализуем форму для генерации пароля с помощью языка гипертекстовой разметки html, каскадных стилевых таблиц css и скриптового языка программирования js.

### Требования к генератору паролей:

Пароль должен содержать заглавные буквы латинского алфавита.

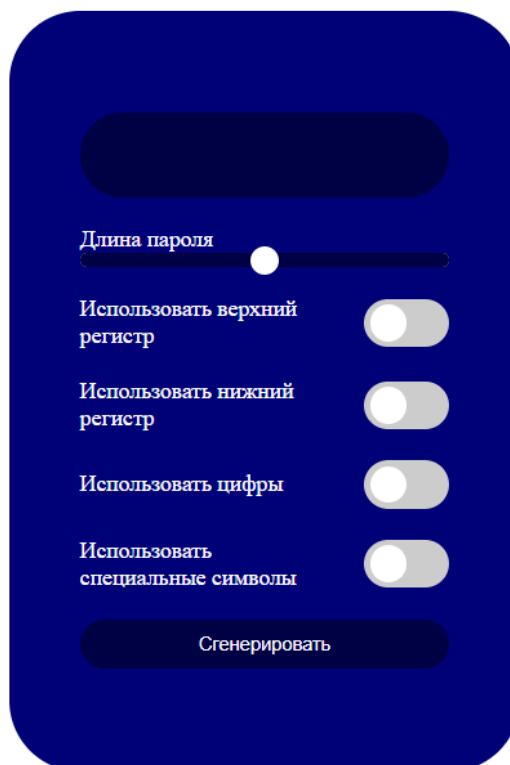
Должны присутствовать строчные буквы латинского алфавита.

Включаются цифры.

Используются специальные символы.

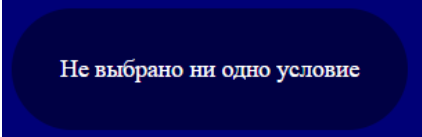
Можно задать длину пароля.

Интерфейс формы выглядит следующим образом: поле для отображения пароля, ползунок длины пароля (минимальная длина пароля 4 символа, максимальная 16 символов), 4 чекбокса для выбора условий пароля (использования верхнего и нижнего регистра, использование цифр и использование специальных символов), а также кнопка сгенерировать.



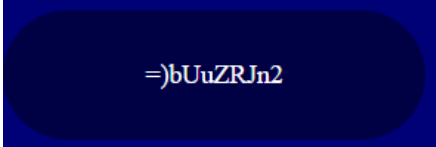
The image shows a UI mockup of a password generator interface. It features a dark blue background with rounded corners. At the top, there is a dark blue rounded rectangle representing the password display field. Below it is a slider control labeled "Длина пароля" (Password length) with a white circular handle. Under the slider are four toggle switches, each with a label: "Использовать верхний регистр" (Use uppercase), "Использовать нижний регистр" (Use lowercase), "Использовать цифры" (Use numbers), and "Использовать специальные символы" (Use special characters). All four toggle switches are currently in the "off" position. At the bottom of the interface is a dark blue rounded rectangle button labeled "Сгенерировать" (Generate).

При отсутствии выбора условия выводится следующее сообщение.



Не выбрано ни одно условие

В случае выбора какого-либо чекбокса выводим пароль



=)bUuZRJn2

В html разметке создадим контейнер, в котором пропишем все поля ввода.

Поле с ползунком и 4 поля чекбокса.

```
<input type="range" id="range" min="4" max="16">
```

```
<input type="checkbox" id="upper">
```

В java script объявим несколько переменных-ссылок на html-элементы.

Ссылка на поле для пароля и кнопка.

```
let genBtn = document.querySelector('.gen_btn');  
let genPass = document.querySelector('.gen_pass');
```

Ссылки на поля ввода.

```
let genRange = document.querySelector('#range');  
let genUpper = document.querySelector('#upper');  
let genLower = document.querySelector('#lower');  
let genNumber = document.querySelector('#number');  
let genSpecial = document.querySelector('#special');
```

Для кнопки создадим обработчик событий при нажатии.

```
genBtn.addEventListener('click', () => {  
  
})
```

Обнуляем текстовые переменные, где chars – строка с набором символов для генерации пароля, а password – сам пароль.

```
let chars = '';  
let password = '';
```

Далее идёт проверка условий, какие типы символов были включены пользователем. В зависимости от выбранного варианта проверки, в строку chars добавляется соответствующий набор символов.

```
if (genUpper.checked) {  
    chars += 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';  
}
```

Проверка наличия выбранных символов. Если пользователь ничего не выбрал, показывается сообщение о необходимости выбрать условия.

```
if (chars == '') {  
    genPass.textContent = 'Не выбрано ни одно условие';  
}
```

Иначе используя цикл, который повторяется столько раз, сколько указал пользователь длиной с помощью ползунка. Каждый раз выбирается случайный символ из массива возможных символов (chars), и он добавляется к строке пароля с помощью функции charAt.

```
for (let i = 0; i < genRange.value; i++) {  
    password += chars.charAt(Math.floor(Math.random() * chars.length));  
}
```

Полученный пароль записывается в элемент вывода genPass.

```
genPass.textContent = password;
```