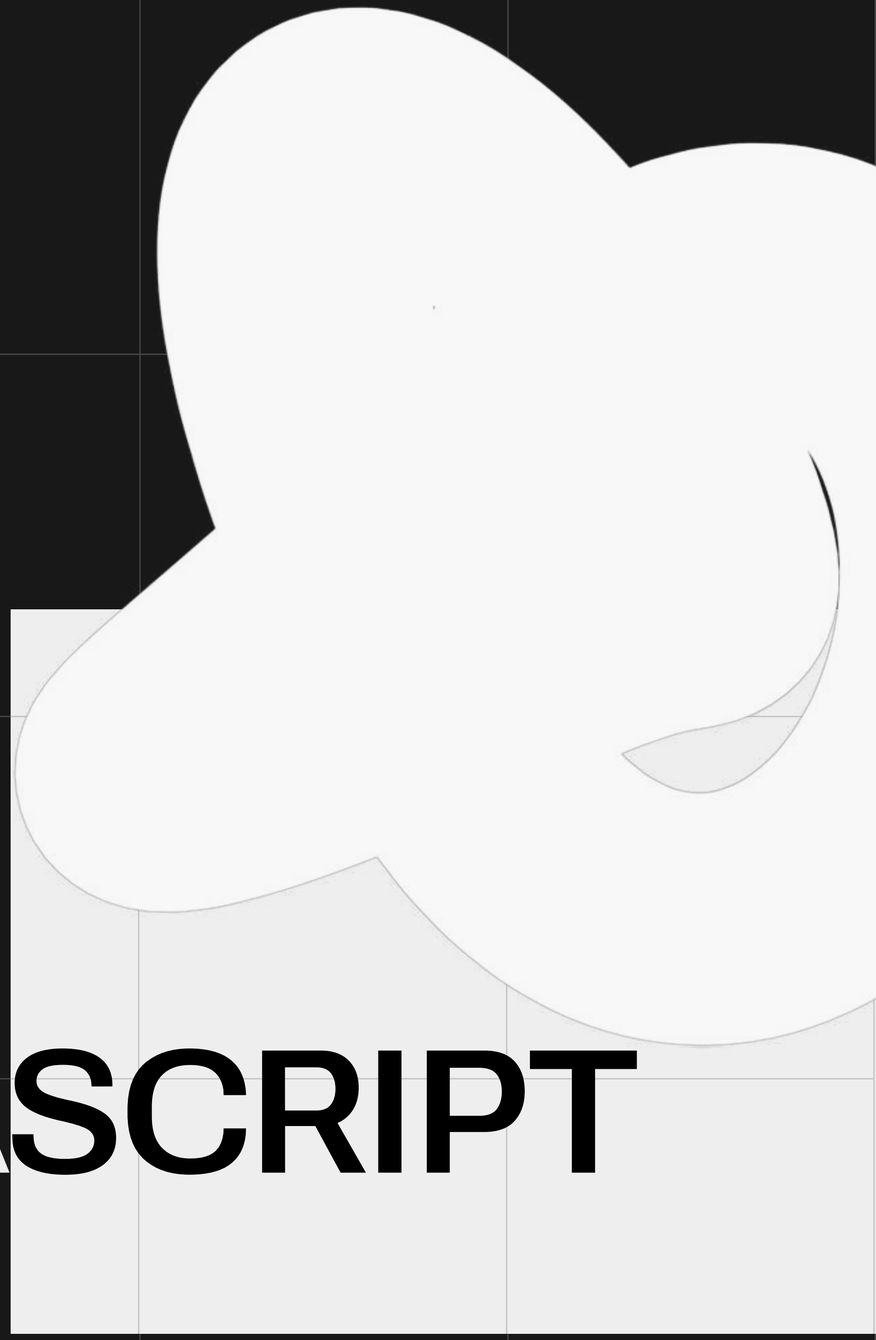
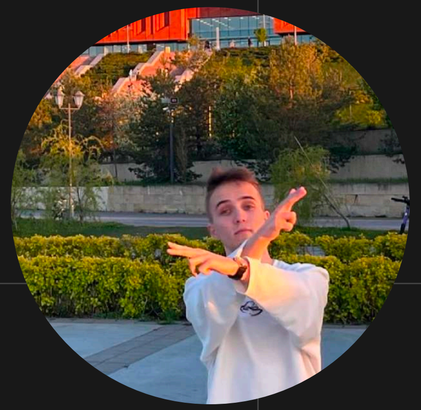


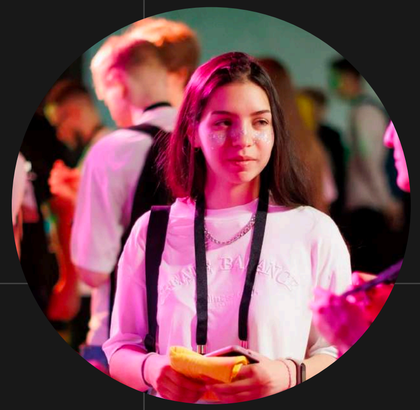
ВВЕДЕНИЕ В JAVASCRIPT

КОМАНДА



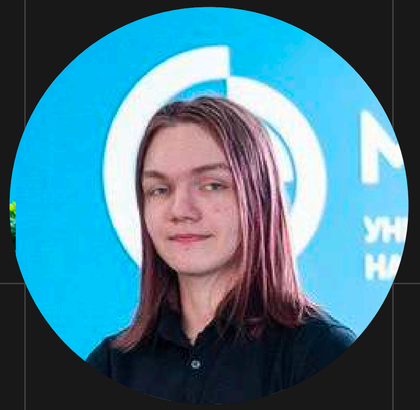
Ярослав Осокин

@osyarik



Егорова Кристина

@kris_bermud



Вадим Солоьев

@vdmkkk

Дмитрий Катрушенко

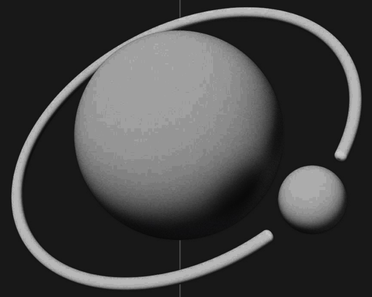
@ClayenKitten

Владислав Сердешнов

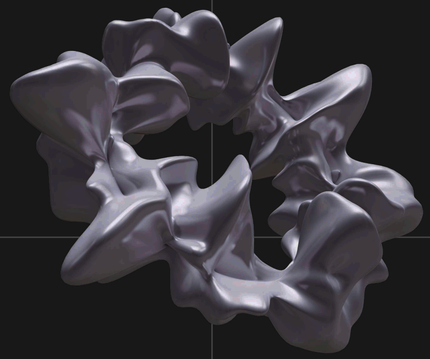
@Serdeshnow

it.am

Синтаксис

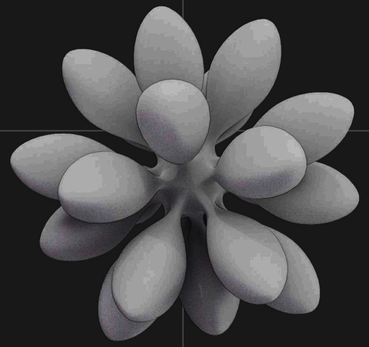


Логические конструкции



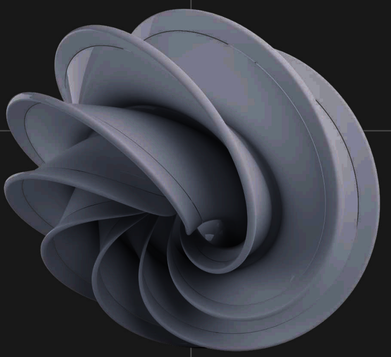
Массивы

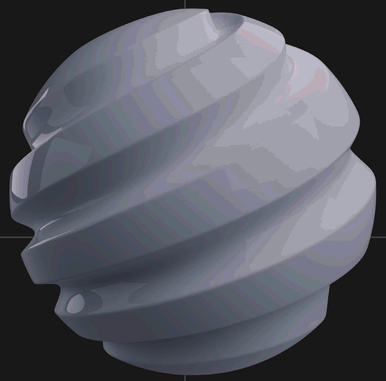
Типы данных



Циклы

Объекты





Функции

it.am

СИНТАКСИС JAVASCRIPT

СИНТАКСИС JAVASCRIPT

it.am

```
// Это однострочный комментарий (ctrl + /)

/*
 Это
 могострочный
 комментарий
*/

let a; // можно ставить точки с запятой
let b // а можно не ставить
const c = 1;
var d; // deprecated
```

СИНТАКСИС JAVASCRIPT

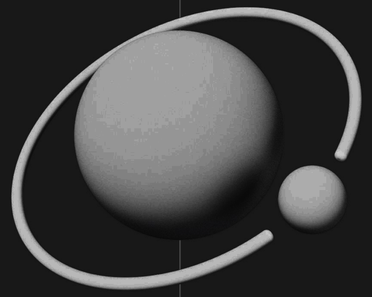
it.am

```
// import используется для импорта ссылок на значения, экспортированные из внешнего
модуля

import defaultExport from "way/to/module";
// будет ссылаться на значение экспорта по умолчанию
import { export1, export2 } from "../../way/to/module";
// будет ссылаться на указанные значения экспорта
import * as name from "../way/to/module";
// будет ссылаться на все экспортируемые значения


// console.log() выводит в консоль браузера заданное в параметре функции сообщение.

console.log( значение );
```

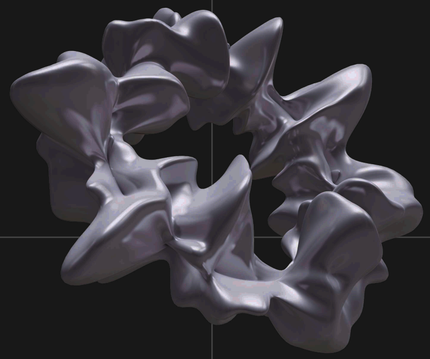
ваши вопросы

it.am

Синтаксис



Логические конструкции

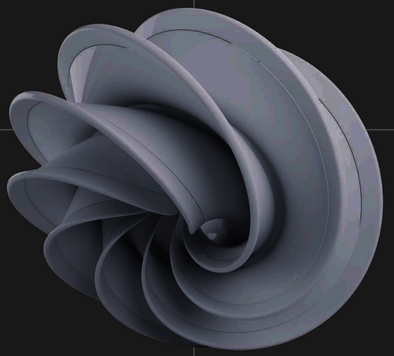


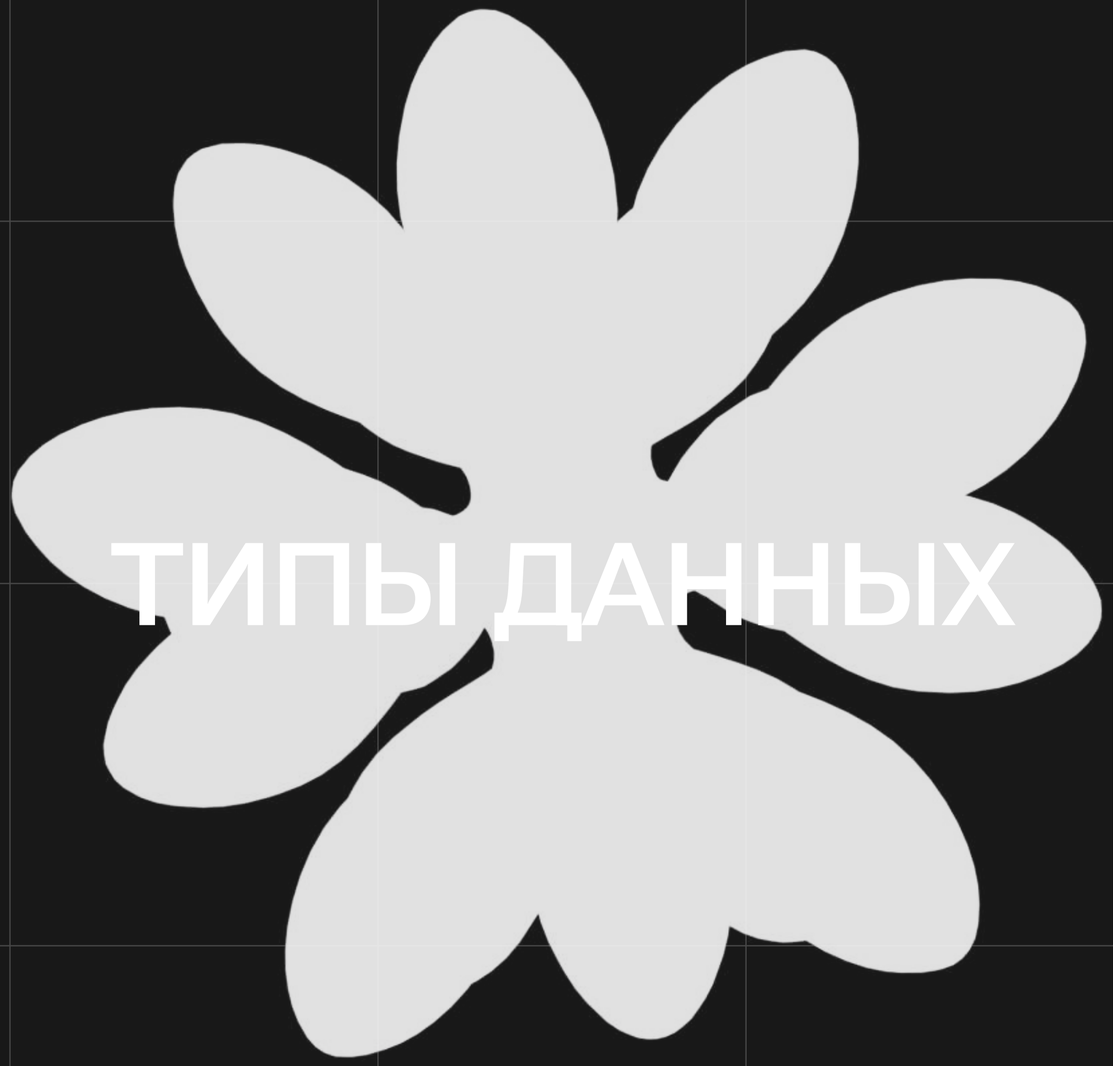
Массивы

Типы данных

Циклы

Объекты





ТИПЫ ДАННЫХ

ТИПЫ ДАННЫХ - СТРОКИ (STRING)

it.am

```
const a = "Строка в одинарных ковычках";

const b = "Строка в двойных ковычках";

const c = `Строка в косых ковычках`;

const d = 'Люблю почитать "Коммерстантъ" вечером у костра';

const e = "Обожаю пряники от \"Яшкино\"!";
```

ТИПЫ ДАННЫХ - ЧИСЛА (NUMBER)

it.am

```
let age = 23;

let weight = 70.1;

let temperature = -5;
```

ТИПЫ ДАННЫХ - БУЛЕВЫЙ (BOOLEAN)

it.am

```
let isNetworking = true;

const hasHomework = false;
```

ТИПЫ ДАННЫХ - ПУСТОЙ (NULL)

it.am

```
// У нас была машина
let myCarName = "Toyota Corolla";

// Но потом мы её продали и остались без машины
myCarName = null;
```

ТИПЫ ДАННЫХ - НЕОПРЕДЕЛЕННЫЙ (UNDEFINED)

```
/*
 Он показывает, что переменная была объявлена,
 но значение ей не присвоено.
*/

let x;
console.log(x); // Выведет undefined*
```

ТИПЫ ДАННЫХ - СИМВОЛ (SYMBOL)

it.am

```
/*
 Используется для создания уникальных ключей объект
а.
 Для создания переменных этого типа существует
 специальная одноименная функция:
*/

const id = Symbol("id");
```

ТИПЫ ДАННЫХ - БОЛЬШОЕ ЧИСЛО (BIGINT)

it.am

```
/*
Используется для хранения очень больших целых чисел
(больше, чем (`2⁵³-1`), т.е. >9007199254740991).
Для создания переменной этого типа нужно поставить б
укву n в конце числа:
*/

let reallyBigNumber = 123456789n;
```

ТИПЫ ДАННЫХ - ОБЪЕКТ (OBJECT)

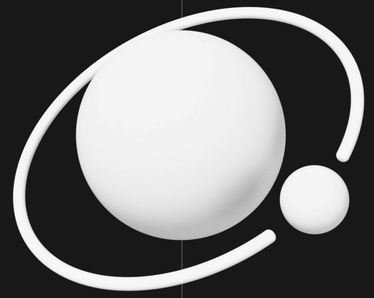
it.am

```
/*Object объявляется с помощью фигурных скобок,
внутри которых следует список пар "key" : value
*/

const car = {
  name: "Toyota Corolla",
  year: 2017,
  isNew: false,
  owner: null,
};
```

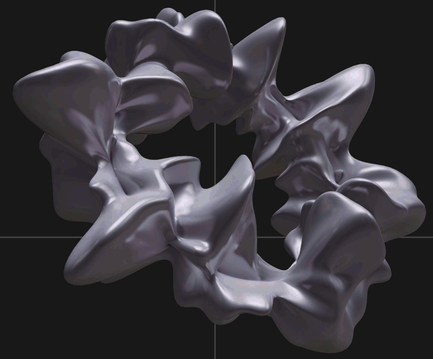
- Строки (String)
- Числа (Number)
- Булевый (Boolean)
- Пустой (Null)
- Неизвестный (Undefined)
- Символ (Symbol)
- Большое Число (BigInt)
- Объект (Object)

it.am

Синтаксис



Логические конструкции

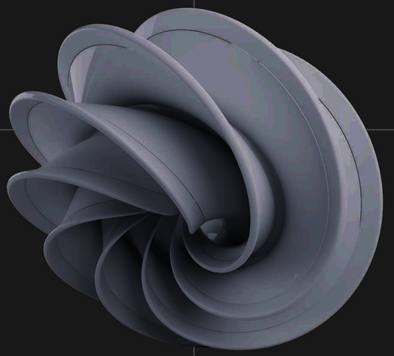


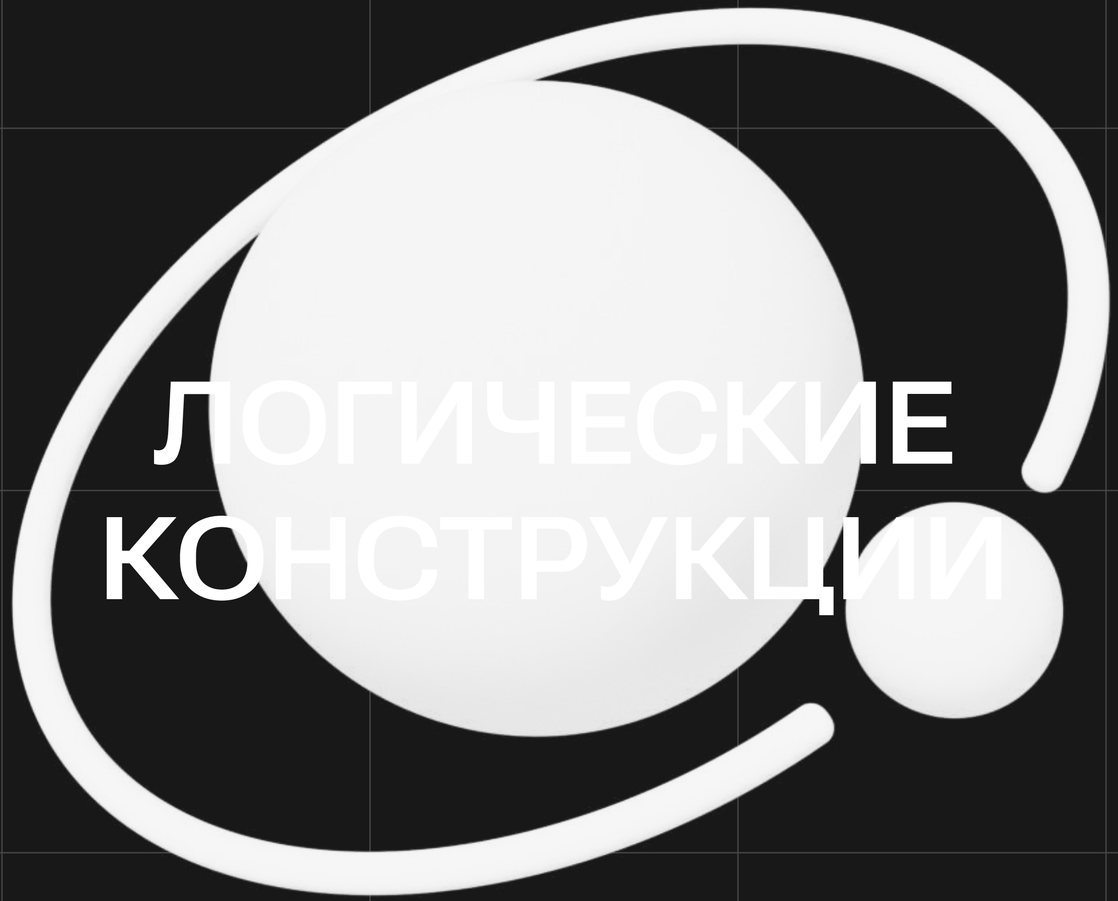
Массивы

Типы данных

Циклы

Объекты





ЛОГИЧЕСКИЕ КОНСТРУКЦИИ

ЛОГИЧЕСКИЕ КОНСТРУКЦИИ - IF

```
const number = 5;

// Передаем выражение
if (number < 10) {
  // Выполнится, так как 5 < 10 === true
  console.log(`Число ${number} меньше 10!`);
}

const string = "";
if (string.length > 0) {
  // Не выполнится
  console.log("Строка содержит текст!");
}

if (string.length) {
  // Не выполнится
  console.log("Строка содержит текст!");
}
```

ЛОГИЧЕСКИЕ КОНСТРУКЦИИ - ELSE

```
const condition = 5 > 10; // вернет false
if (condition) {
  console.log("Выражение истино!");
} else {
  console.log("Выражение ложно!");
}
// Вывод: Выражение ложно!

const string = "";
if (string.length) {
  console.log("Строка содержит текст!");
} else {
  console.log("Строка пустая!");
}
// Вывод: Строка пустая!
```

ЛОГИЧЕСКИЕ КОНСТРУКЦИИ - ELSE IF

```
const status = "online";
if (status === "online") {
  console.log("Ваш статус: в сети");
} else if (status === "offline") {
  console.log("Ваш статус: не в сети");
} else {
  console.log("Ошибка! Неизвестный статус!");
}
Вывод: Ваш статус: в сети
```

ЛОГИЧЕСКИЕ КОНСТРУКЦИИ - ELSE IF - SVELTE

```
  let count = 0;
</script>

{#if count > 10}
  <p>{count} is less than 5</p>
{:else if count < 5}
  <p>{count} is greater than 10</p>
{:else}
  <p>{count} is between 5 and 10</p>
{/if}

<style lang="css">
  p {
    font-size: 1rem;
  }
</style>
```

ЛОГИЧЕСКИЕ КОНСТРУКЦИИ - SWITCH

```
switch (переменная) {
  case значение1:
    // выполнится, если переменная === значение1
    break; // завершение случая
  case значение2:
    // выполнится, если переменная === значение2
    break;
  default:
    // выполнится, если ни один другой случай не сработал
    break;
}
```

ЛОГИЧЕСКИЕ КОНСТРУКЦИИ - SWITCH

```
const status = "online";
switch (status) {
  case "online":
    console.log("Ваш статус: в сети");
    break;
  case "offline":
    console.log("Ваш статус: не в сети");
    break;
  default:
    console.log("Ошибка! Неизвестный статус!");
    break;
}
```

ЛОГИЧЕСКИЕ КОНСТРУКЦИИ - ТЕРНАРНЫЙ ОПЕРАТОР

it.am

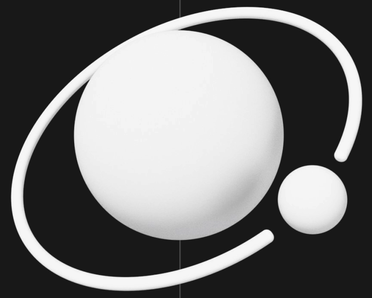
```
const num = 7;
console.log(num === 7 ? "Число 7" : "Число не 7"); // 'Число 7'
console.log(num === 10 ? "Число 10" : "Число не 10"); // 'Число не 10'

// Запись в переменную
const text = num > 42 ? "Число больше 42" : "Число меньше 42";
console.log(text); // 'Число меньше 42'
```

it.am

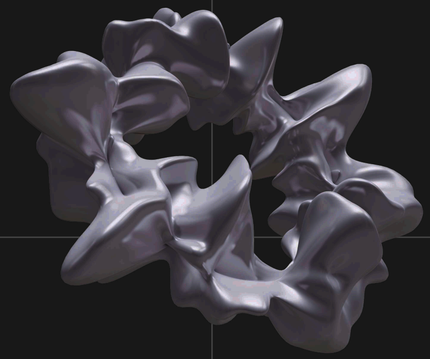
- if
- else
- else if
- switch
- Тернарный оператор

it.am

Синтаксис



Логические конструкции

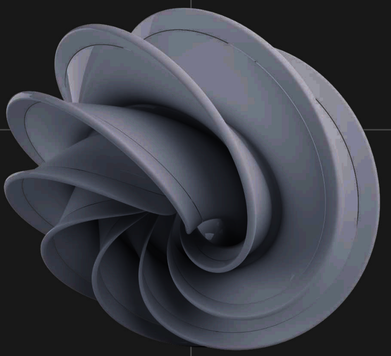


Массивы

Типы данных

Циклы

Объекты




ЦИКЛЫ

ЦИКЛЫ - WHILE

it.am

```
let i = 0;
while (i < 5) {
  console.log(i);
  i++;
}

/*
  Вывод:
  0
  1
  2
  3
  4
*/
```

ЦИКЛЫ - WHILE

it.am

```
let i = 0;
while (i < 5) {
  console.log(i);
  i++;
}

/*
  Вывод:
  0
  1
  2
  3
  4
*/
```

ЦИКЛЫ - WHILE

it.am

```
let i = 5;
while (i) {

// приведение типов в условии цикла
  console.log(i);
  i--;
}
/*
  Вывод:
  5
  4
  3
  2
  1
*/
```

ЦИКЛЫ - DO ... WHILE

```
let i = 0;
do {
  console.log(i);
  i++;
} while (i < 5);

/*
  Вывод:
  0
  1
  2
  3
  4
*/
```

ЦИКЛЫ - FOR

```
for (let i = 0; i < 5; ++i
) {onsole.log(i);
}

/*
  Вывод в консоли:
  0
  1
  2
  3
  4
*/
```

ЦИКЛЫ - FOR

it.am

```
const str = 'Hello';
for (let i = 0; i < str.length; i++) {
  console.log(str[i]);
}
/*
  Вывод:
  H
  e
  l
  l
  o
*/
```

ЦИКЛЫ - BREAK

it.am

```
let i = 0;
while (i < 10) {
  console.log(i);

  if (i === 3) break;

  i++;
}

/*
  Выведет:
  0
  1
  2
  3
*/
```

ЦИКЛЫ - CONTINUE

it.am

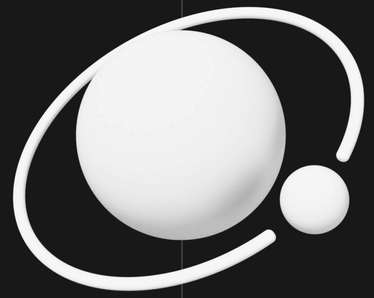
```
for (let i = 0; i < 5; ++i
) {
  if (i == 2) {
    continue;
  }
  console.log(i);
}

/*
  Вывод в консоли:
  0
  1
  3
  4
*/
```

it.am

- while
- do ... while
- for
- break
- continue

it.am

Синтаксис



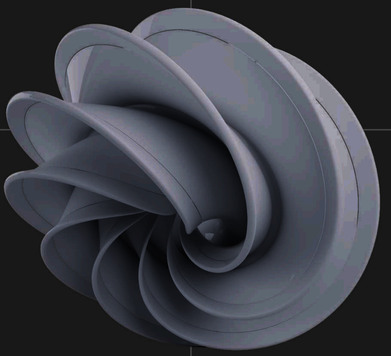
Логические конструкции

Массивы

Типы данных

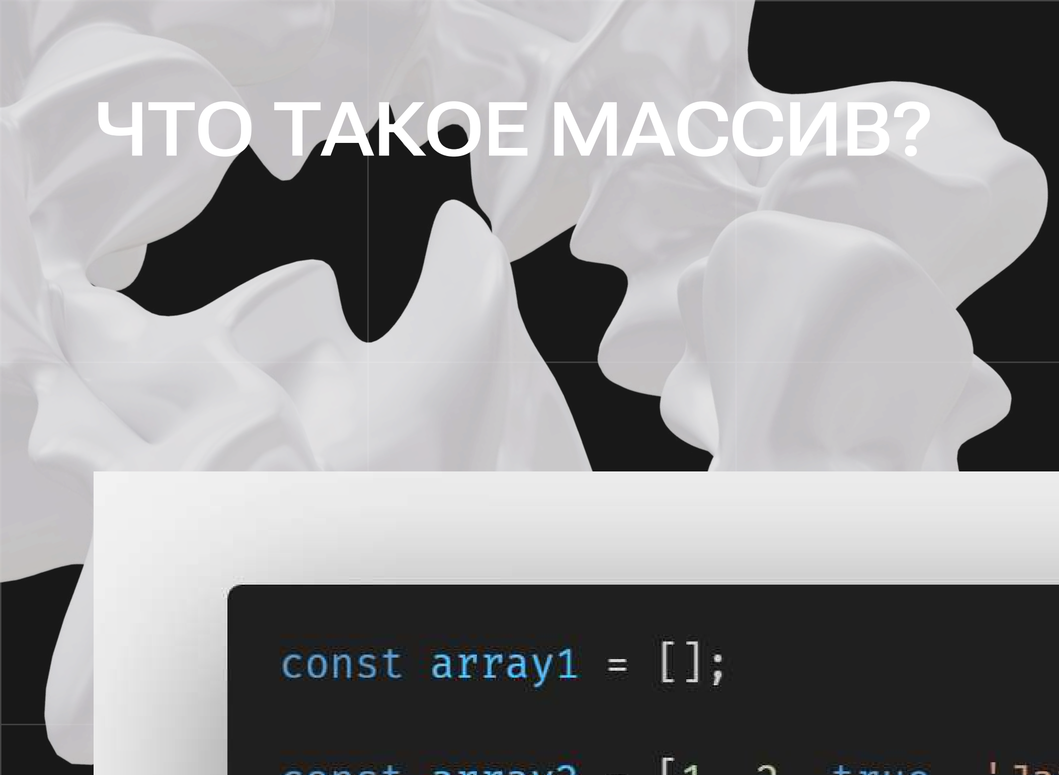
Циклы

Объекты



МАССИВЫ

ЧТО ТАКОЕ МАССИВ?



```
const array1 = [];

const array2 = [1, 2, true, 'JavaScript', 'Hello World!'];
```

МАССИВ - ПОЛУЧЕНИЕ ЭЛЕМЕНТОВ

it.am

```
const array = ["first", "second", "third", "fourth"];

console.log("item:", array[1]); // item: ???

console.log("item:", array[4]); // item: ???
```

МАССИВ - ПОЛУЧЕНИЕ ЭЛЕМЕНТОВ СВОЙСТВО LENGTH

it.am

```
const array = ["first", "second", "third", "fourth"];

console.log("last item:", array[array.length - 1]);
// last item: fourth

console.log("second item from end:", array[array.length - 2]);
// second item from end: third
```

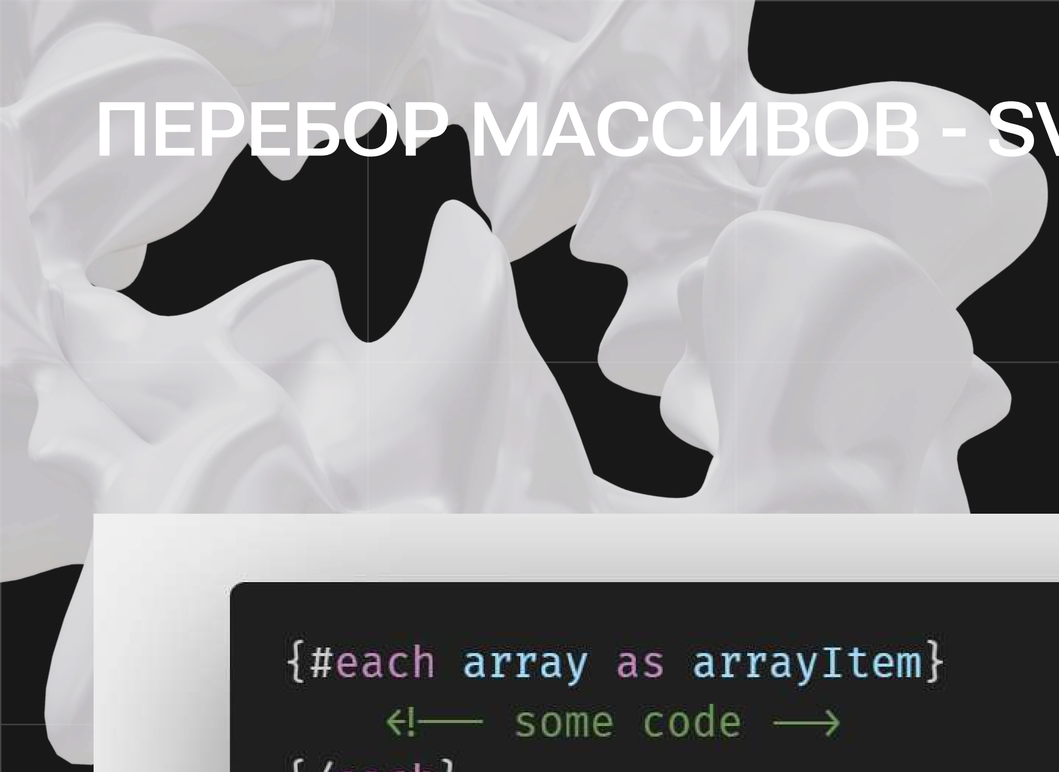
ПЕРЕБОР МАССИВОВ - FOR

```
const favouriteFruits = ["apple", "pear", "banana", "plum"];

for (let i = 0; i < favouriteFruits.length; ++i) {
  console.log(fruit);
}
```

ПЕРЕБОР МАССИВОВ - FOR ... OF

```
const favouriteFruits = ["apple", "pear", "banana", "plum"];

for (fruit of favouriteFruits) {
  console.log(fruit);
}
```

ПЕРЕБОР МАССИВОВ - SVELTE



```
{#each array as arrayItem}
    <!-- some code -->
{/each}
```

ПЕРЕБОР МАССИВОВ - SVELTE

it.am

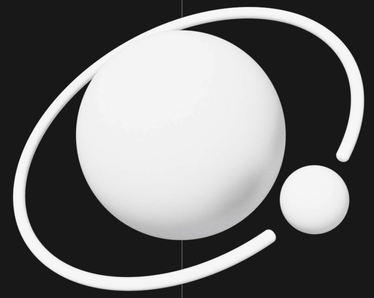
```svelte
<script lang="js">
const favouriteFruits = ["apple", "pear", "banana", "plum"];
</script>

<h2>Мои любимые фрукты:</h2>
<ul>
  {#each favouriteFruits as fruit}
  <li>{fruit}</li>
  {/each}
</ul>

<style lang="css">
  li {
    font-style: italic;
  }
</style>
```

it.am

- Массив (Array)
- Работа Массивов
- Перебор Массивов

it.am

Синтаксис



Логические конструкции

Массивы

Типы данных

Циклы

Объекты


ОБЪЕКТЫ

ЧТО ТАКОЕ ОБЪЕКТ?

it.am

```
// Пустой объект
const empty = {};

const car = {
  name: "VolksWagen Polo",
  year: 2017,
  isNew: false,
};
```

ЧТО ТАКОЕ ОБЪЕКТ?

it.am

```
const car = {
  name: "VolksWagen Polo",
  year: 2017,
  isNew: false,
  owner: {
    name: "Ivanov Ivan",
    experience: 5,
  },
};
```

ОБЪЕКТ И ЕГО СВОЙСТВА

it.am

```
const car = {
  name: "VolksWagen Polo",
  year: 2017,
  isNew: false,
};

console.log(car.name); // VolksWagen Polo
// Или передаем название ключа в кавычках (это строка)
console.log(car["name"]); // VolksWagen Polo
```

ОБЪЕКТ - СОЗДАНИЕ СВОЙСТВ

it.am

```
const car = {
  name: "Toyota Corolla",
};

const key = "color";
car[key] = "red";

console.log(car);
```

ОБЪЕКТ - ОБНОВЛЕНИЕ СВОЙСТВ

it.am

```
const car = {
  name: "Toyota Corolla",
};

// Создание свойств
car.engine = 1.6;
car["maxSpeed"] = 185;

// Обновление свойства
car.name = "My Car";

console.log(car);
// { name: 'My Car', engine: 1.6, maxSpeed: 185 }
```

ОБЪЕКТ - УДАЛЕНИЕ СВОЙСТВ

```
const car = {
  name: "Toyota Corolla",
  engine: 1.6,
  maxSpeed: 185,
};

delete car.engine;
delete car.maxSpeed;

console.log(car);
// { name: 'Toyota Corolla' }
```

ПЕРЕБОР ОБЪЕКТОВ

it.am

```
const car = {
  name: "Toyota Corolla",
  year: 2017,
  isNew: false,
};

for (const key in car) {
  console.log(`${key}: ${car[key]}`);
}

// name: Toyota Corolla
// year: 2017
// isNew: false
```

ОПРЕАТОР SPREAD

it.am

```
const car = {
  name: "Toyota Supra",
  year: 2017,
};

const carOwner = {
  ownerName: "Ivan Ivanov",
  ownerAge: 27,
};

const carFullInfo = {
  ...car,
  ...carOwner,
  additionalInfo: "Эта лекция когда-нибудь закончится?",
};

console.log(carFullInfo);
```

ОБЪЕКТ - СЛИЯНИЕ СВОЙСТВ

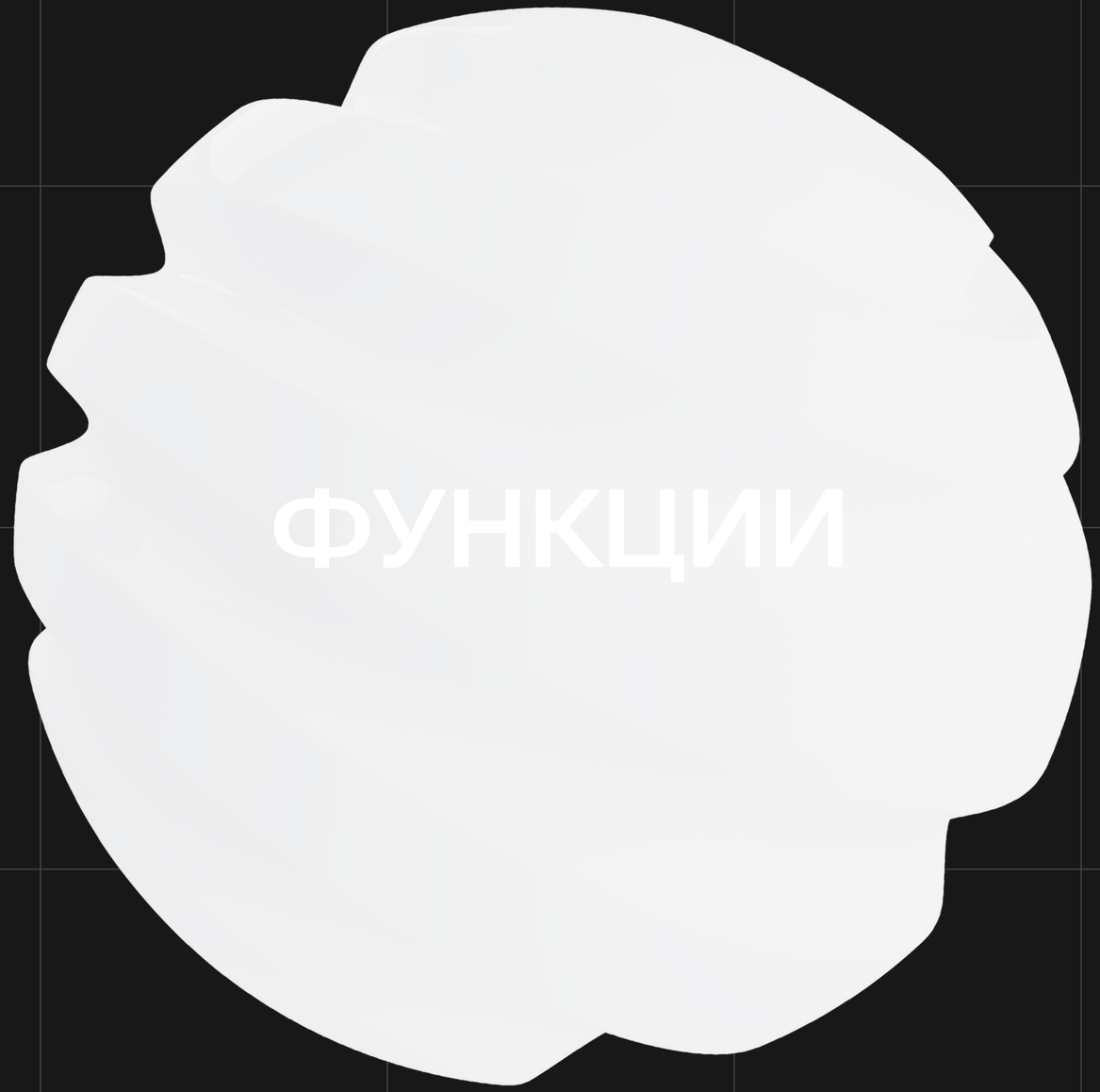
it.am

```
const obj1 = { x: 1, y: 1 };
const obj2 = { x: 2, y: 2 };

const unitedObj = {
  ...obj1,
  ...obj2,
  y: 3,
};

console.log(unitedObj.x); // Выведет 2
console.log(unitedObj.y); // Выведет 3
```

- Объект (Object)
- Свойства объекта
- Объект - ссылочный тип
- Перебор объектов
- Оператор spread

it.am

Функции


ФУНКЦИИ

ЧТО ТАКОЕ ФУНКЦИЯ?

it.am

```
const tax = 0.2;

const price1 = 100;
const count1 = 5;
const totalPrice1 = price1 * count1 * (1 + tax); // Вычисляем итоговую цену

const price2 = 350;
const count2 = 2;
const totalPrice2 = price2 * count2 * (1 + tax); // Повторяем вычисления опять

// И так далее...
```

ФНУКЦИЯ - СИНТАКСИС

it.am

```
function doSomething() {
  // Код функции
}
```

ФНУКЦИЯ - ПАРАМЕТРЫ

it.am

```
function showTotalPrice(price, count) {
  const totalPrice = price * count; // Используем параметры в вычислениях
  console.log(totalPrice);
}
```

ФНУКЦИЯ - ПАРАМЕТРЫ

it.am

```
function showTotalPrice(price, count) {
  console.log(price * count);
}

showTotalPrice(100, 5); // Выведет в консоль: 500
showTotalPrice(350, 2); // Выведет в консоль: 700
```

ФНУКЦИЯ - ВОЗВРАТ ЗНАЧЕНИЯ

it.am

```
function getTotalPrice(price, count) {
  const tax = 0.2;
  return price * count * (1 + tax); // Возвращаем результат
}

const totalPrice1 = getTotalPrice(100, 5); // Запишет: 600
const totalPrice2 = getTotalPrice(350, 2); // Запишет: 840
```

FUNCTION DECLARATION

it.am

```
const myCat = catToHumanAge(5);

function catToHumanAge(realAge) {
  console.log("Input cat age:", realAge);
  let message = '';

  if(realAge <= 1) {
    message = 'your cat\'s human age is under 15'
  }
  else if (realAge == 2) {
    message = 'your cat\'s human age is 24'
  }
  else {
    let age = 24 + ((realAge - 2) * 4);
    message = `your cat's human age is ${age}`
  }

  return message;
}
```

FUNCTION EXPRESSION

it.am

```
const doSomething = function () {
  // Код функции
};
```

FUNCTION EXPRESSION

it.am

```
const addTen = function (number) {
  const result = number + 10;
  return result
};

console.log(addTen(23));
```

ARROW FUNCTION EXPRESSION

it.am

```
const sayHello = (name) ⇒ alert(`Hello, ${name}`);
const calc = (a, b) ⇒ a + b;

sayHello("Mark"); // Выведет сообщение "Hello, Mark"
console.log(calc(1, 2)); // Выведет в консоль: 3
```

ФУНКЦИИ - ТРИ ВИДА

it.am

```
function doSomething() { /* ... */ } // Function Declaration

const doSomething = function () { /* ... */ }; // Function Expression

const doSomething = () => { /* ... */ }; // Arrow Function Expression
```

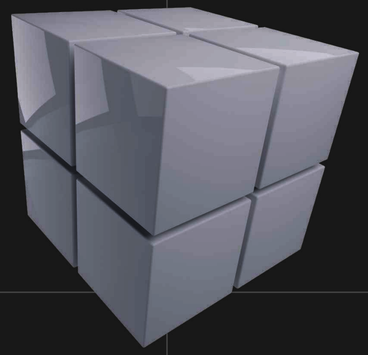
it.am

- Функция
- Параметры функции
- Ключевое слово return

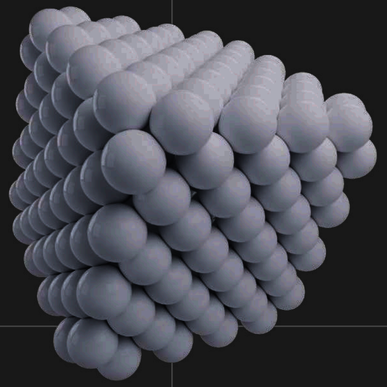
- Function Declaration
- Function Expression
- Arrow Function Expression

СПАСИБО ВАМ!

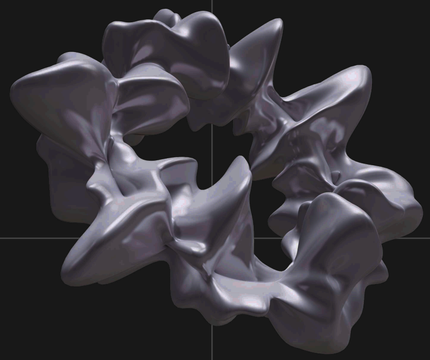
it.am



Математика
JavaScript

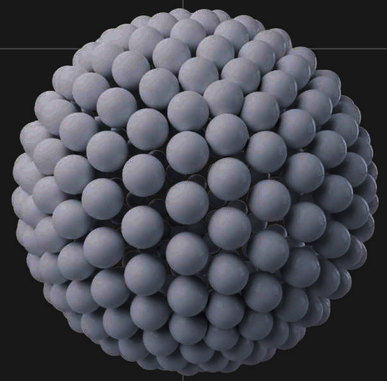


Логические
Операторы



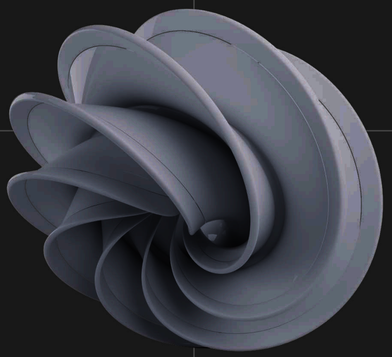
Массивы

Операторы
сравнения

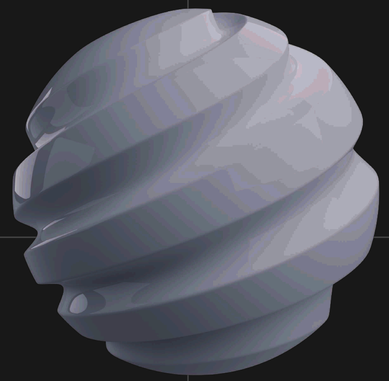


Методы
Строк

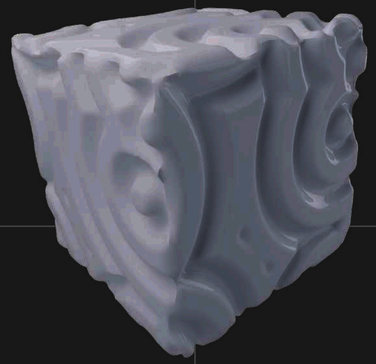
Объекты



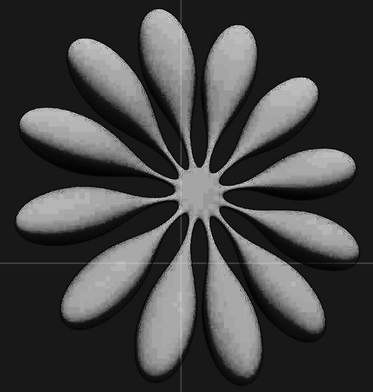
it.am



Функции



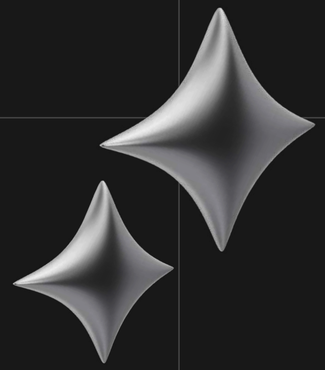
Шаблоны



События

Svelte

Компоненты



QR - CODES


PIZZA TIME.