Модули

Модули

Модули в ES6

- *Модулем считается файл с кодом. В этом файле ключевым словом export помечаются переменные и функции, которые могут быть использованы снаружи. Другие модули могут подключать их через вызов import*

- Целью модулей es6 было создание формата, удобного как для пользователей CJS, так и для пользователей AMD. В связи с этим они имеют такой же компактный синтаксис, как и модули CJS. С другой стороны, они не такие динамичные как и AMD

Особенности

- На этапе компиляции возникнет ошибка, если попытаться импортировать что-то, что не было предварительно экспортировано
- Можно легко осуществить асинхронную загрузку модулей
- Декларативный синтаксис (для импорта и экспорта)
- Программная загрузка (чтобы задать конфигурацию загрузки модулей и для условной загрузки модулей)

Пример

```
// lib/math.js
export function sum (x, y) { return x + y; }
export var pi = 3.141593;

// index.js
import * as math from "lib/math";
console.log("2π = " + math.sum(math.pi,
math.pi));

// other.js
import { sum, pi } from "lib/math";
console.log("2π = " + sum(pi, pi));
```

При запуске в chrome видим ошибку Uncaught SyntaxError: Unexpected token import. Слабая поддержка браузеров. Необходимо использовать babel.js либо аналогичный инструмент

Export

1. export { name1, name2, …, nameN };
2. export { variable1 as name1, variable2 as name2, …, nameN };
3. export let name1, name2, …, nameN;//или var, const
4. export let name1 = …, name2 = …, …, nameN;//или var, const
5. export default выражение;
6. export default function (…) { … } //или class, function*
7. export default function name1(…) { … } //или class, function*
8. export { name1 as default, … };
9. export * from …;
10. export { name1, name2, …, nameN } from …;
11. export { import1 as name1, import2 as name2, …, nameN } from …;

Export переменных

- возможно перед объявлением переменных, функций и классов
- возможно отдельно, при этом в фигурных скобках указывается то, что именно экспортируется

```
export let
one = 1;
```

```
let two = 2;
export {two};
```

```
export
{one,
two};
```

Ключевое слово AS

При AS можно указать, что переменная one будет доступна снаружи (экспортирована) под другим именем .
Пример:

```
export {one as once, two as twice};
```

Export функций и классов

```
1.  export class User {
2.      constructor(name) {
3.          this.name = name;
4.      }
5.  };
6.  export function sayHi() {
7.      console.log("Hello!");
8.  };
9.  //export {User, sayHi}
```

Экспорт по умолчанию

Иногда модуль экспортирует только одно значение (класс). В таком случае удобно определить это значение как экспортируемое по умолчанию

```
1.  //user.js
2.  export default class User {
3.      constructor(name) {
4.          this.name = name;
5.      }
6.  };
7.  //login.js:
8.  import User from './user';
9.  new User("Петя");
```

Import

Другие модули могут подключать экспортированные значения при помощи ключевого слова import

1. import defaultMember from "module-name";
2. import * as name from "module-name";
3. import { member } from "module-name";
4. import { member as alias } from "module-name";
5. import { member1 , member2 } from "module-name";
6. import { member1 , member2 as alias2 , [...] } from "module-name";
7. import defaultMember, { member [, [...]] } from "module-name";
8. import defaultMember, * as name from "module-name";
9. import "module-name";

Пример

```
1. //nums.js:
2. export let one = 1;
3. export let two = 2;
4. //1.js
5. import {one, two} from "./nums";
6. console.log(`${one} and ${two}`);//1 and 2
7. //2.js
8. //использование as
9. //импорт one под именем item1, а two – под именем item2
10. import {one as item1, two as item2} from "./nums";
11. console.log(`${item1} and ${item2}`);//1 and 2
```

Импорт всех значений в виде объекта

1. import * as numbers from "./nums";
2. //теперь экспортированные переменные - свойства numbers
3. console.log(`${numbers.one} and ${numbers.two}`);//1 and 2

Встроенные модули

```
1. //ES5
2. <script>
3.     (function () {
4.  /*начало
   самовызывающейся
   функции*/
5.         var tmp = 100;
6. //не станет глобальной
7.     }());
8. //конец самовызывающейся
9. //функции
10. </script>
```

```
1. //ES6
2. <script>
3.     module {
4. /*анонимный
   внутренний модуль*/
5.         let tmp = 100;
6. //не станет глобальной
7.     }
8. </script>
```

Импорт модулей и загрузка скриптов

```
1. System.import(['module1', 'module2'],
2.    function (module1, module2) { //успешное выполнение … },
3.    function (err) { //ошибка … } );
```

Отец: Сынок, почему у тебя глаза красные?

Сын: Я курил траву

Отец: Не ври мне, ты плакал потому, что кодишь на JavaScript