JS Импорт внешних модулей

Модульные системы

Поддерживаются модульные системы ES6 (ECMAScript 6) и CommonJS. Код узла является корневым модулем системы ES6. В модулях ES6 поддерживается статический и динамический импорт модулей ES6, для импорта модулей CommonJS применяется функция `require` (см. Полное описание API).

▼ Примеры

Модульная система ECMAScript 6

Внешний модуль "./foo/module/module.js":

```
function sayHello() {
    return "Hello";
}
export { sayHello };
```

Внешний модуль "./foo/foo.js":

```
import { sayHello } from "./module/module.js";
function cube(x) {
    return x * x * x;
}
export { cube, sayHello };
```

Использование модулей в коде узла JavaSript:

```
// Статический импорт
import { cube, sayHello } from "foo/foo.js";
console.log(sayHello());
console.log('3^3 = ', cube(3));

// Динамический импорт
import("foo/foo.js").then(mod => {
     console.log(mod.sayHello());
     console.log('3^3 = ', mod.cube(3));
}).catch(e => {
     console.error(e);
});
// или
(async () => {
     try {
        const mod = await import("foo/foo.js");
        console.log(mod.sayHello());
        console.log('3^3 = ', mod.cube(3));
     } catch(e) {
        console.error(e);
     }
})();
```

Модульная система CommonJS

Внешний модуль "./foo/module/module.js":

```
function sayHello() {
    return "Hello";
}
module.exports = sayHello;
```

Внешний модуль "./foo/foo.js":

```
const sayHello = require("./module/module.js");
function cube(x) {
    return x * x * x;
}
exports.sayHello = sayHello;
exports.cube = cube;
```

Использование модулей в коде узла JavaSript:

```
const foo = require("foo/foo.js");
console.log(foo.sayHello());
console.log('3^3 = ', foo.cube(3));
```

Внешние модули могут быть в кодировке `UTF-8` или `UTF-16 Little Endian с BOM`.

Работа модулей CommonJS реализована в соответствии со спецификацией, за следующими отличиями:

- Необязательные свойства `require.main`, `require.paths` и `module.uri` отсутствуют;
- Добавлены необязательные свойства `require.resolve` и `require.cache` (как в NodeJS);
- Объект модуля имеет свойства `parent`, `loaded` и `filename` (как в NodeJS);
- Из модуля доступна глобальная переменная `__filename`, хранящая абсолютный путь к модулю внутри файлового хранилища.

▼ Пример

Внешний модуль "child_module.js":

```
console.log("Hello! I am " + __filename);
exports.filename = module.filename; // возвращает полный путь к
child_module.js
exports.parent = module.parent.id;  // возвращает идентификатор
вызывающего модуля
exports.loaded = module.loaded;     // возвращает true или false - был
ли загружен модуль
```

Использование модуля "child_module.js" в коде узла JavaSript:

```
// require.resolve:
// - на сервере Loginom возвращает полный путь модуля в файловом
хранилище
// - в настольной редакции возвращает полный путь модуля в файловой
системе
let path = require.resolve("child_module.js");
console.log(path);
// Вызов внешнего модуля системы CommonJS
let childModule = require("child_module.js");
console.log(childModule.filename);
console.log(childModule.parent);
console.log(childModule.loaded);
// Очищается кэш модуля "child_module.js"
delete require.cache[path];
// и внешний модуль вызывается повторно,
// в результате чего повторно выводится "Hello! I am ... ".
// Без очистки кэша этого не происходит
require("child_module.js");
```

Из модулей CommonJS можно загружать только модули CommonJS.

Пути импорта внешних модулей

Пути импорта модулей могут быть относительными или абсолютными.

Относительный путь

При использовании относительных путей в корневом модуле (скрипте, который содержит редактор кода узла):

- Если пакет сохранен, то относительный путь к внешнему модулю указывается от расположения пакета.
- Если пакет не сохранен, то относительный путь к внешнему модулю указывается от каталога пользователя.

Во внешних модулях относительный путь указывается от расположения модуля, осуществляющего импорт.

Пример:

```
import { cube, foo, sayHello } from "./foo/foo.js";
```

Абсолютный путь

На сервере Loginom

Если путь начинается с "/", то он интерпретируется как абсолютный внутри файлового хранилища и считается от каталога пользователя.

Пример:

```
import { cube, foo, sayHello } from "/user/foo/foo.js";
```

В настольной редакции

Абсолютный путь — полный путь в файловой в системе. Пример:

```
import { cube, foo, sayHello } from "C:\\Script\\foo\\foo.js";
```