

# Eralink JS SDK

## Назначение библиотеки

Библиотека предоставляет методы для взаимодействия с Eralink.

## Установка библиотеки

Библиотека предоставляется заказчику по запросу.

## Конфигурация

```
{
  "webSocketUrLs": [
    "ws://GENA:9000/websocket",
    "ws://GENB:9000/websocket"
  ],
  "webSocketSendTimeout": 5,
  "maxReconnectAttempt": 3,
  "reconnectTimeout": 10,
  "agent": {}, // необязательный параметр
  "listeners": {}, // необязательный параметр
  "strings": {} // необязательный параметр
}
```

### webSocketUrLs

Массив websocket-адресов

### webSocketSendTimeout

Таймаут отправки сообщений (в секундах)

### maxReconnectAttempt

Максимальное кол-во попыток восстановления соединения при ошибке подключения к Eralink ChatServer по WebSocket

### reconnectTimeout

Таймаут повторного подключения к Eralink ChatServer по WebSocket (в секундах)

### agent

Устанавливает агента (вместо передачи объекта можно воспользоваться методом `setAgent` )

```
{
  id: string; // id агента/оператора
  domain: string; // например "genesys"
  resource: string; // например "wwe"
}
```

## listeners

Устанавливает слушатели `on`, `off`, `trigger` (при подключении к `WWE` нужно передать объект `Backbone.Events`)

```
{
  on?: (eventName?: EralinkEventName, callback?: any, context?: any) => void;
  off?: (eventName?: EralinkEventName, callback?: any, context?: any) => void; // Выполняется при закрытии
  WebSocket-соединения
  trigger?: (eventName: EralinkEventName, event: any) => void; // Ловит события при установлении/закрытии
  соединения, отправке/получении сообщений
}
```

## strings

Устанавливает объект строк (информационные сообщения, ошибки), которые передаются во вне. Необходим для локализации.

Дефолтный объект (файл `strings.ts`):

```
{
  browser_not_support_websocket_error: "Web browser does not support Websocket",
  websocket_is_not_valid_error: "Websocket url could not be defined or is not valid",
  request_param_invalid_error: "Request param invalid",
  websocket_invalid_state_error: "Websocket has invalid state",
  failed_to_connect_websocket_error: "Failed to connect to WebSocket, error: <%= error %> . Next attempt will be in
  <%= timeout %> seconds",
  response_has_not_returned_error: "Response with id [<%= reqId %>] has not returned while timeout",
  unable_to_send_message_error: "Unable to send message: invalid parameters"
}
```

## Создание класса методов

Библиотека содержит класс `EralinkProtocol`, который нужно импортировать и создать, передав объект конфигурации:

```
var eralink = new EralinkProtocol({
  websocketUrls: config.websocketUrls,
  websocketSendTimeout: config.websocketSendTimeout,
  maxReconnectAttempt: config.maxReconnectAttempt,
  reconnectTimeout: config.reconnectTimeout
})
```

## Методы класса `EralinkProtocol`

### `setAgent(agent)`

Метод устанавливает отправителя. Принимает объект:

```
{
  id: string; // id агента/оператора
}
```

```
domain: string; // например "genesys"  
resource: string; // например "wwe"  
}
```

## connect()

Метод устанавливает WebSocket-соединение.

## disconnect()

Метод закрывает WebSocket-соединение.

## sendPresence(partyTo, isUnsubscribe, eralinkMessageId)

Метод отправляет Presence. Принимает аргументы:

partyTo: XmppUser - принимает объект данных пользователя, с которым нужно установить соединение.

```
// XmppUser  
{  
  id: string; // id получателя в eralink  
  domain: string; // домен пользователя, например "eralink"  
  resource: string; // ник пользователя, например "Иванов Иван"  
}
```

isUnsubscribe: boolean - принимает true или false. Если true, то presence.type устанавливается как unavailable  
eralinkMessageId?: string - id последнего сообщения (необязательный аргумент)

## sendMessage(message)

Метод отправляет сообщение. Принимает объект:

```
{  
  message: {  
    id?: string, // id сообщения (необязательный параметр)  
    timestamp?: number, // время отправки сообщения в миллисекундах (необязательный параметр)  
    to: {  
      id: string, // id получателя в eralink  
      domain: string // домен пользователя, например "eralink"  
    },  
    type: string, // например "chat", "groupchat", "error"  
    body?: string, // текст сообщения (необязательный параметр)  
    extensions: [] // массив вложений (image, file, sticker, audio, location, contact, video)  
  }  
}
```

## sendText(data, to, from)

Метод отправляет текстовое сообщение. Принимает аргументы:

data: string - текст сообщения.

to: XmppUser - объект данных получателя сообщения.

from?: XmppUser - объект данных отправителя сообщения (необязательный аргумент, если не передать from, объект отправителя будет браться из this.agent который задается методом setAgent или в конфигурации).

```
// XmppUser
{
  id: string; // id получателя в eralink
  domain: string; // домен пользователя, например "eralink"
  resource: string; // ник пользователя, например "Иванов Иван"
}
```

## sendFile(data, to, from)

Метод отправляет файлы без сжатия. Принимает аргументы:

`data: File` - объект файла.

`to: XmppUser` - объект данных получателя сообщения.

`from?: XmppUser` - объект данных отправителя сообщения (необязательный аргумент, если не передать `from`, объект отправителя будет браться из `this.agent` который задается методом `setAgent` или в конфигурации).

```
// XmppUser
{
  id: string; // id получателя в eralink
  domain: string; // домен пользователя, например "eralink"
  resource: string; // ник пользователя, например "Иванов Иван"
}
```

## sendImage(data, to, from)

Метод отправляет картинку (не путать с `sendFile`, данный файл будет сжат мессенджером/соц. сетью). Принимает аргументы:

`data: File` - объект файла картинки.

`to: XmppUser` - объект данных получателя сообщения.

`from?: XmppUser` - объект данных отправителя сообщения (необязательный аргумент, если не передать `from`, объект отправителя будет браться из `this.agent` который задается методом `setAgent` или в конфигурации).

```
// XmppUser
{
  id: string; // id получателя в eralink
  domain: string; // домен пользователя, например "eralink"
  resource: string; // ник пользователя, например "Иванов Иван"
}
```

## sendVideo(data, to, from)

Метод отправляет видео (не путать с `sendFile`, данный файл будет сжат мессенджером/соц. сетью). Принимает аргументы:

`data: File` - объект видео-файла.

`to: XmppUser` - объект данных получателя сообщения.

`from?: XmppUser` - объект данных отправителя сообщения (необязательный аргумент, если не передать `from`, объект отправителя будет браться из `this.agent` который задается методом `setAgent` или в конфигурации).

```
// XmppUser
{
  id: string; // id получателя в eralink
  domain: string; // домен пользователя, например "eralink"
}
```

```
resource: string; // ник пользователя, например "Иванов Иван"  
}
```

### **sendLocation(data, to, from)**

Метод отправляет локацию. Принимает аргументы:

`data: LocationExtension` - объект локации.

`to: XmppUser` - объект данных получателя сообщения.

`from?: XmppUser` - объект данных отправителя сообщения (необязательный аргумент, если не передать `from`, объект отправителя будет браться из `this.agent` который задается методом `setAgent` или в конфигурации).

```
// LocationExtension  
{  
  lat: number; // широта  
  lon: number; // долгота  
}  
  
// XmppUser  
{  
  id: string; // id получателя в eralink  
  domain: string; // домен пользователя, например "eralink"  
  resource: string; // ник пользователя, например "Иванов Иван"  
}
```