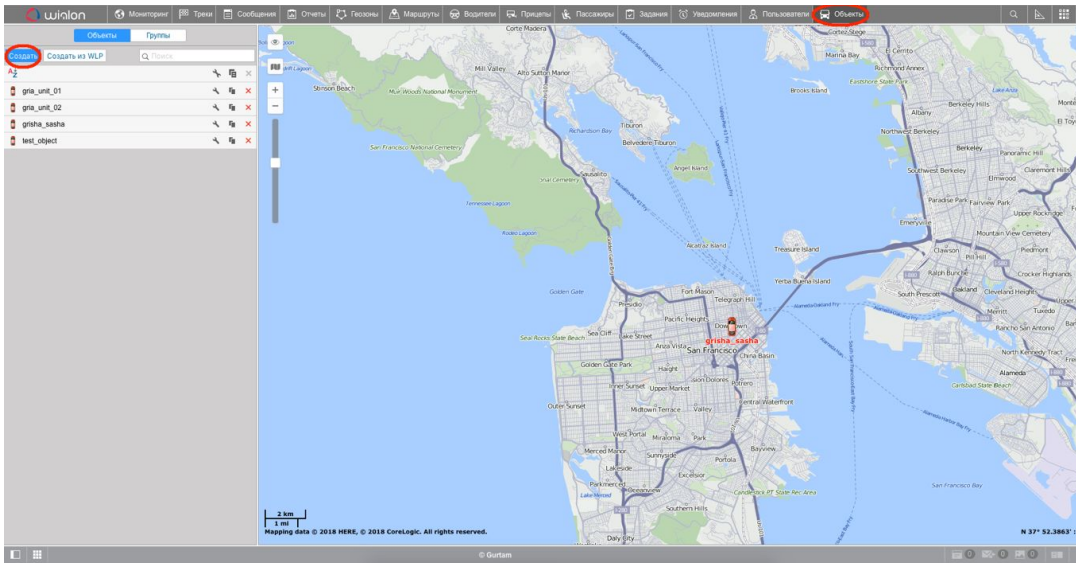
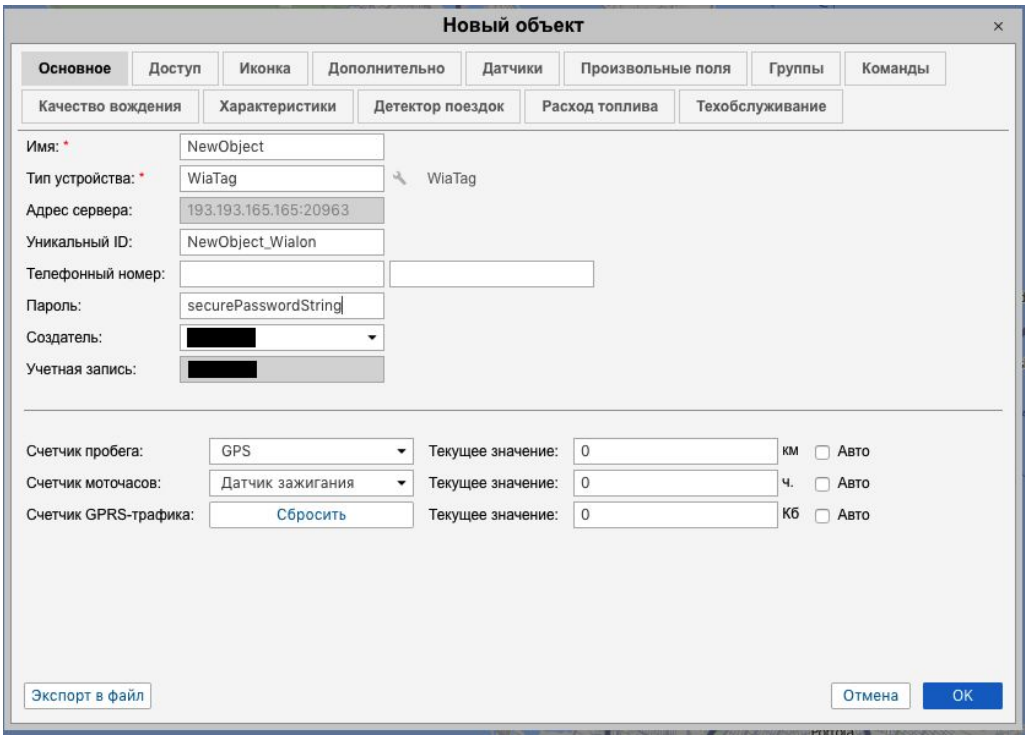
iOS Manual

1. Create the unit that will send messages to Wialon: access your user account, open the **“Units”** tab, press the **“New”** button.



2. Set up the unit in the popped-up dialog: specify the **Name**, select the WiaTag **Device type**, fill in the **Unique ID** and **Password** fields.

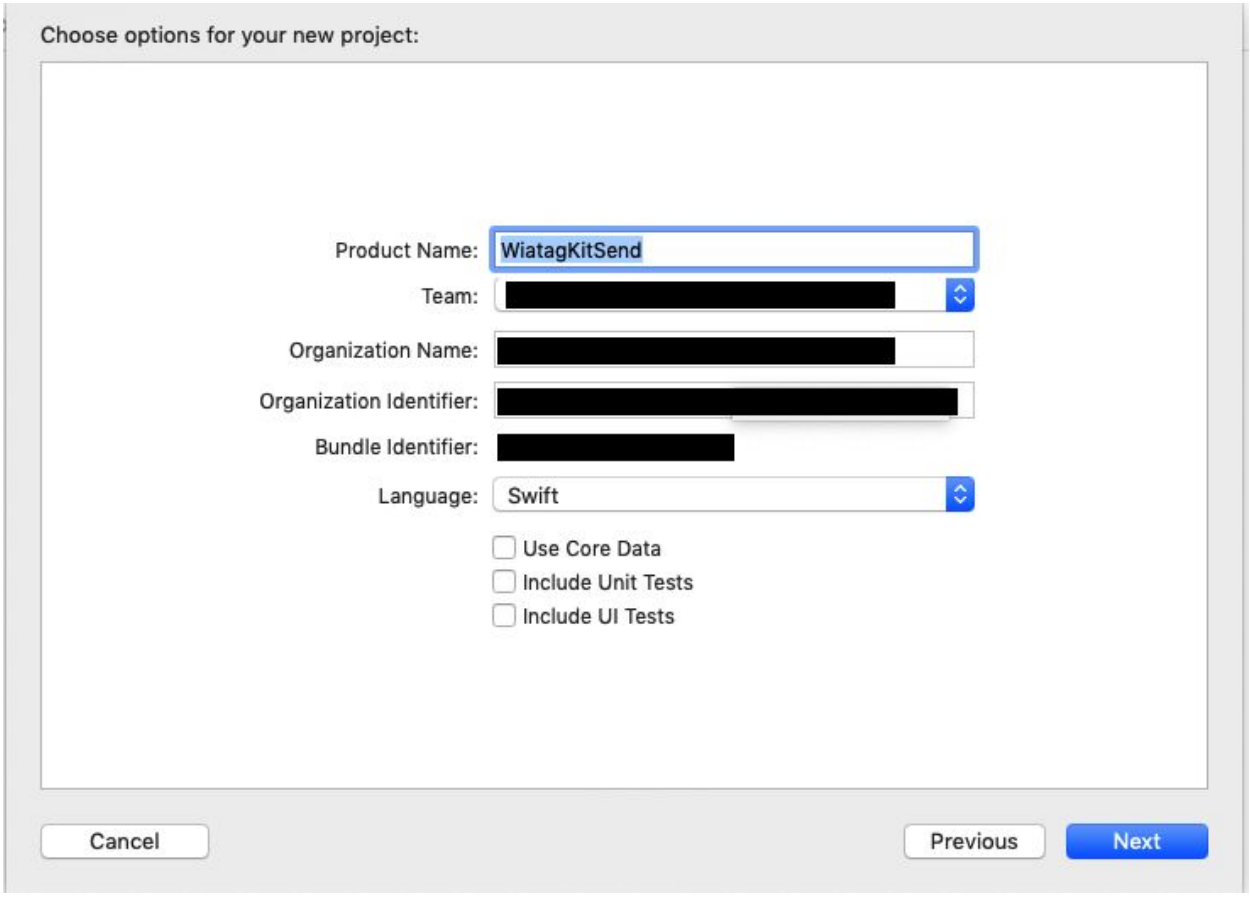
Note! Anyone with the unit’s ID and password can send messages to Wialon on behalf of this unit. Make sure that only authorized people know the password.



To send the message, you will need the **Server address**, **Unique ID** and **Password** fields.

Lets have some coding

Work in **Xcode 10** environment, select **Swift 4.2** language. Create a Single-view App with the WiatagKitSend name.



Connect **wiatag-kit-ios** with the help of **cocoapods**. To do this, we execute in the terminal:

```
cd /path/to/your/project/
pod init
```

Then open **Podfile** and paste there the lines:

```
pod 'WiaTagKit'
```

Get back to the terminal and execute:

```
pod install
```

Close the project and open **workspace**. Move to **ViewController.swift** and import the **WiaTagKit** module.

Sending message to Wialon

1. Initialize **WTMessageSender** using the Wialon unit's **Server Address**, **Unique ID** and **Password**.
2. Initialize the **WTMessage** unit.
3. Send **WTMessage** using the corresponding **WTSender** method.

Here is the example of **ViewController** realization that sends an empty message when the controller is shown:

```
import UIKit
import WiaTagKit

class ViewController: UIViewController {

    override func viewDidAppear(_ animated: Bool) {
        super.viewDidAppear(animated)
        sendMessage()
    }

    func sendMessage() {
        let host = "193.193.165.165"
        let port: UInt = 20963
        let unitId = "NewObject_Wialon"
        let password = "securePasswordString"
        let sender = WTMessageSender(host: host,
                                     port: port,
                                     unitId: unitId,
                                     password: password)
        let message = WTMessage { builder in }
        sender.send(message) { error in
            guard let error = error else {
                print("message sending completed with success")
                return
            }
            print("message sending failed with error \(error)")
        }
    }
}
```

Copy this code and insert it into your **ViewController.swift.**

Note! The **host**, **port**, **unitId** and **password** values should be replaced by your WiaTag unit's **Server Address (IP, port)**, **Unique ID,** and **Password**.

Launching the project
If you did everything right, the message: **message sending completed with success** appears in the Xcode console.

It means we sent an empty message.

Add the content to the message to WiaTag

Currently, the message can contain the following information:

1. **Time of the message creation**. It is sent anyways but can be redefined, means it may differ from the actual time of the message creation.
2. **Location**. Most likely you will want to send it if your app works with the **CoreLocation** service. Two constructor methods are used in this case:
   `WTLocation(location: CLLocation)` – this constructor is likely to be enough for you.
   `WTLocation(latitude: Double, longitude: Double, altitude: Double, speed: UInt16, bearing: UInt16, satellites: UInt8).`
3. **SOS-message flag**.
4. **Image**. The WTImage class is used to send the image.
5. **Text message**.
6. **Battery charge level**.
7. **Parameters**. They have the keys (only strings) and the values (text, binary values and the ones of the **Int**, **Long**, **Float**, **Double** types). You can't send 2 parameters for one and the same value of the key.

For example, let's create the message that sends time, SOS-signal, image, text message, and Int-parameter:

```
let message = WTMessage { builder in
            //setup time
            let date = Date(timeIntervalSinceNow: -20)
            builder.time = date
            //setup location
            let location = CLLocation(latitude: 53, longitude: 27)
            builder.location = WTLocation(location: location)
            //setup image
            let image = UIImage(named: "free_image.jpg")
            let imageData = image?.jpegData(compressionQuality: 1)
            if let imageData = imageData {
                builder.image = WTImage(imageData: imageData,
                                        named: "imageName.jpg")
            }
            //setup SOS signal
            builder.isSos = true
            //setup text message
```

```
            builder.text = "This is my text message!"
            //setup int param
            builder.addParam("int value", withIntValue: 3)
        }
```

You can send all messages with one method call. It is handier when you need to send several messages.

```
func sendMessages() {
        var messages = [WTMessage]()

        let host = "193.193.165.165"
        let port: UInt = 20963
        let unitId = "NewObject_Wialon"
        let password = "securePasswordString"
        let sender = WTMessageSender(host: host,
                                     port: port,
                                     unitId: unitId,
                                     password: password)

        for i in 0...10 {
            let message = WTMessage { builder in
                //setup time
                let date = Date(timeIntervalSinceNow: TimeInterval(-i * 10))
                builder.time = date
                //setup location
                let location = CLLocation(latitude: 53, longitude: 27)
                builder.location = WTLocation(location: location)
                //setup image
                let image = UIImage(named: "free_image.jpg")
                let imageData = image?.jpegData(compressionQuality: 1)
                if let imageData = imageData {
                    builder.image = WTImage(imageData: imageData,
                                            named: "imageName.jpg")
                }
                //setup SOS signal
                builder.isSos = true
                //setup text message
                builder.text = "This is my \(i) text message!"
                //setup int param
                builder.addParam("int value", withIntValue: 3)
            }
```

```
            messages.append(message)
        }

        sender.send(messages) { error in
            guard let error = error else {
                print("message sending completed with success")
                return
            }
            print("message sending failed with error \(error)")
        }
    }
```

It is one of the examples of how to work with **WiaTagKit.** Feel free to ask any questions on the new library usage contacting us at development@gurtam.com.