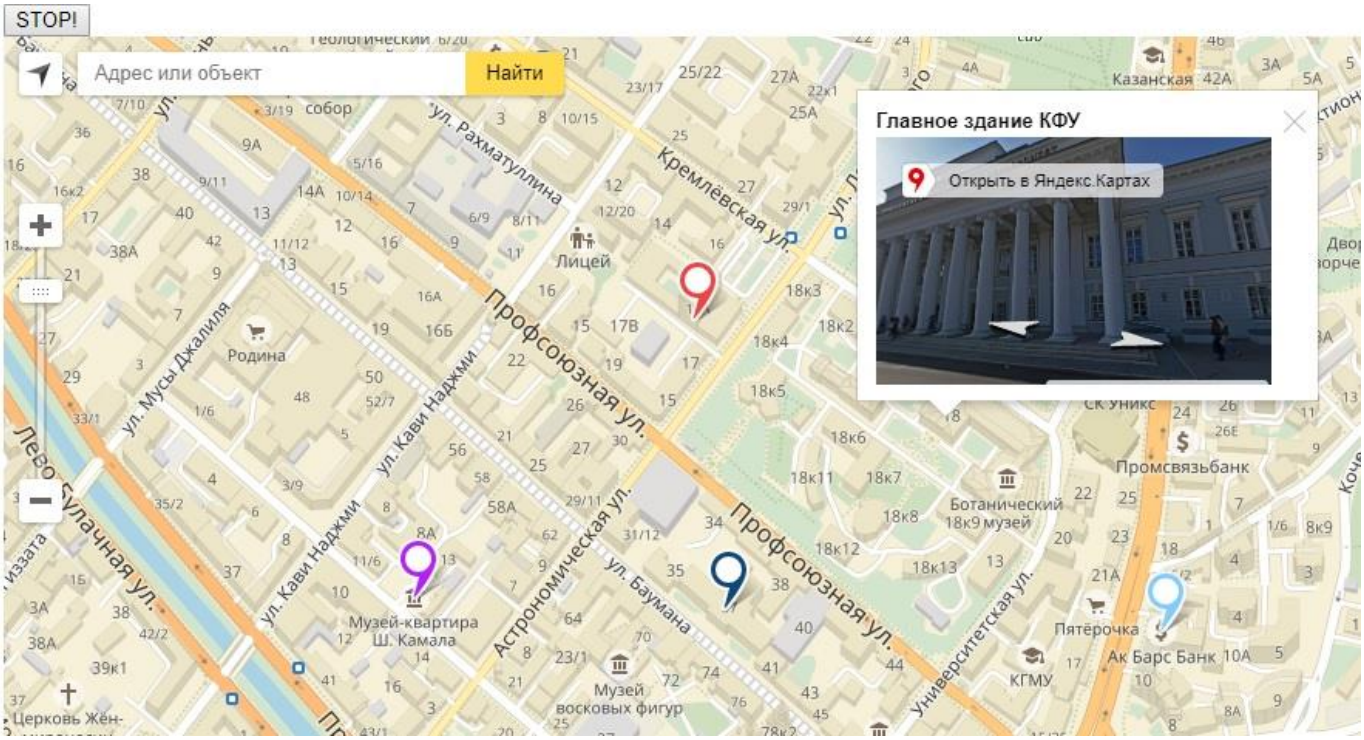
Казанский (Приволжский) Федеральный университет
Институт Вычислительной математики и информационных технологий

Отчёт по работе «API Яндекс.Карт»

Студент 09-641 Десятов Александр

Казань – 2017

Скриншот:



Файл icon_customImage.js

```
ymaps.ready(init);

function init() {
    var myMap = new ymaps.Map("map", {
        center: [55.79, 49.12],
        zoom: 17
    }, {
        searchControlProvider: 'yandex#search'
    }),
        myPlacemark1 = new ymaps.Placemark([55.791521, 49.118164], {
            name: 'Институт физики КФУ',
            dir: [17, 35], //развернуть панораму по координатам дома
            panoLayer: 'yandex#panorama'
        }, {
            preset: 'islands#redIcon',
            openEmptyBalloon: true,
            balloonPanelMaxMapArea: 0
        }),
        myPlacemark2 = new ymaps.Placemark([55.7907520, 49.121649], {
            name: 'Главное здание КФУ',
            dir: [197, 10],
            panoLayer: 'yandex#panorama'
        }, {
            preset: 'islands#greenIcon',
            openEmptyBalloon: true,
            balloonPanelMaxMapArea: 0
        });
```

```
myPlacemark3 = new ymaps.Placemark([55.78925894, 49.118586159], {
    name: 'Национальный банк',
    dir: [130, 10],
    panoLayer: 'yandex#panorama'
}, {
    preset: 'islands#nightIcon',
    openEmptyBalloon: true,
    balloonPanelMaxMapArea: 0
});
myPlacemark4 = new ymaps.Placemark([55.789344988, 49.1142832], {
    name: 'Музей Ш.Камала',
    dir: [180, 10],
    panoLayer: 'yandex#panorama'
}, {
    preset: 'islands#violetIcon',
    openEmptyBalloon: true,
    balloonPanelMaxMapArea: 0
});
myPlacemark5 = new ymaps.Placemark([55.78908685, 49.124667], {
    name: 'Ак Барс Банк',
    dir: [130, 10],
    panoLayer: 'yandex#panorama'
}, {
    preset: 'islands#lightblueIcon',
    openEmptyBalloon: true,
    balloonPanelMaxMapArea: 0
});


// В этой функции выполняем проверку на наличие панорамы в данной точке.
function requestForPanorama(e) {
    var placemark = e.get('target'),
        // Координаты точки, для которой будем запрашивать панораму.
        coords = placemark.geometry.getCoordinates(),
        // Тип панорамы (воздушная или наземная).
        panoLayer = placemark.properties.get('panoLayer');

    placemark.properties.set('balloonContent', "Идет проверка на наличие панорамы...");

    // Запрашиваем объект панорамы.
    ymaps.panorama.locate(coords, {
        layer: panoLayer
    }).then(
        function(panoramas) {
            if (panoramas.length) {
                // Устанавливаем для балуна макет, содержащий найденную панораму.
                setBalloonContentLayout(placemark, panoramas[0], placemark.properties.get('dir'));
```

```
            } else {
                // Если панорам не нашлось, задаем
                // в содержимом балуна простой текст.
                placemark.properties.set('balloonContent', "Для данной точки панорамы нет.");
            }
        },
        function(err) {
            placemark.properties.set('balloonContent',
                "При попытке открыть панораму произошла ошибка: " + err.toString());
        }
    );
}
// Функция, устанавливающая для метки макет содержимого ее балуна.
function setBalloonContentLayout(placemark, panorama, direct) {
    // Создание макета содержимого балуна.
    var BalloonContentLayout = ymaps.templateLayoutFactory.createClass(
        '<div id="panorama" style="width:256px;height:180px">' + '<b>{{properties.name}}</b><br />'
+ '$[properties.text]' + '</div>', {
            // Переопределяем функцию build, чтобы при формировании макета
            // создавать в нем плеер панорам.
            build: function() {
                // Сначала вызываем метод build родительского класса.
                BalloonContentLayout.superclass.build.call(this);
                // Добавляем плеер панорам в содержимое балуна.
                this._openPanorama();
            },
            // Аналогично переопределяем функцию clear, чтобы удалять
            // плеер панорам при удалении макета с карты.
            clear: function() {
                this._destroyPanoramaPlayer();
                BalloonContentLayout.superclass.clear.call(this);
            },
            // Добавление плеера панорам.
            _openPanorama: function() {
                if (!this._panoramaPlayer) {
                    // Получаем контейнер, в котором будет размещаться наша панорама.
                    var el = this.getParentElement().querySelector('#panorama');
                    this._panoramaPlayer = new ymaps.panorama.Player(el, panorama, {
                        controls: ['panoramaName'],
                        direction: direct
                    });
                }
            },
            // Удаление плеера панорамы.
            _destroyPanoramaPlayer: function() {
                if (this._panoramaPlayer) {
                    this._panoramaPlayer.destroy();
```

```
                this._panoramaPlayer = null;
            }
        }
    });
    // Устанавливаем созданный макет в опции метки.
    placemark.options.set('balloonContentLayout', BalloonContentLayout);
}

function randomInt(min, max) {
    var rand = min - 0.5 + Math.random() * (max - min + 1)
    rand = Math.round(rand);
    return rand;
}

function randomOpen() {
    rand = randomInt(1, 5);
    if (rand == 1) myPlacemark1.balloon.open();
    else if (rand == 2) myPlacemark2.balloon.open();
    else if (rand == 3) myPlacemark3.balloon.open();
    else if (rand == 4) myPlacemark4.balloon.open();
    else myPlacemark5.balloon.open();
}

function closeBal() {
    if (myPlacemark1.balloon.isOpen())
        myPlacemark1.balloon.close();
    else if (myPlacemark2.balloon.isOpen())
        myPlacemark2.balloon.close();
    else if (myPlacemark3.balloon.isOpen())
        myPlacemark3.balloon.close();
    else if (myPlacemark4.balloon.isOpen())
        myPlacemark4.balloon.close();
    else if (myPlacemark5.balloon.isOpen())
        myPlacemark5.balloon.close();
}


myMap.geoObjects.add(myPlacemark1);
myMap.geoObjects.add(myPlacemark2);
myMap.geoObjects.add(myPlacemark3);
myMap.geoObjects.add(myPlacemark4);
myMap.geoObjects.add(myPlacemark5);

myPlacemark1.events.once('balloonopen', requestForPanorama);
myPlacemark2.events.once('balloonopen', requestForPanorama);
myPlacemark3.events.once('balloonopen', requestForPanorama);
myPlacemark4.events.once('balloonopen', requestForPanorama);
```

```
    myPlacemark5.events.once('balloonopen', requestForPanorama);
        //Получаем ID таймера, чтобы потом его отключить
    timeid = setInterval(randomOpen, randomInt(2, 10) * 1000);
    // Скрываем хинт при открытии балуна.
    myMap.events.add('balloonopen', function(e) {
        myMap.hint.close();
        setTimeout(closeBal, 5000);
    });

}
```

Файл icon_customImage.html

```
<html>

<head>
    <title>Сигнализация</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
//Кнопка, чтобы остановить открытие балунов
        <input type="button" value="STOP!" onclick="clearInterval(timeid);" />
        <script src="http://api-maps.yandex.ru/2.1/?lang=ru_RU" type="text/javascript"></script>
    <script src="icon_customImage.js" type="text/javascript"></script>
    <style>
        html,
        body,
        #map {
            width: 100%;
            height: 100%;
            padding: 0;
            margin: 0;
        }
    </style>
</head>

<body>
    <div id="map">

    </div>

</body>

</html>
```