Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 6

по курсу «Разработка мобильных приложений»

«Интеграция Яндекс.Карт»

Студент группы ИУ9-72Б Шемякин В.А.

Преподаватель Посевин Д. П.

Москва 2025

1 Задание

Реализовать виджет Яндекс.Карт вывода объектов согласно варианта из таблицы ниже, в виджете должна отображаться Яндекс.Карта с расположенными на ней метками объектов, по клику на метку объекта должен открываться виджет с подробной информацией об объекте.

Вариант: $http : //pstgu.yss.su/iu9/mobiledev/lab4_yandex_map/2023.php?x = var15$

2 Результаты

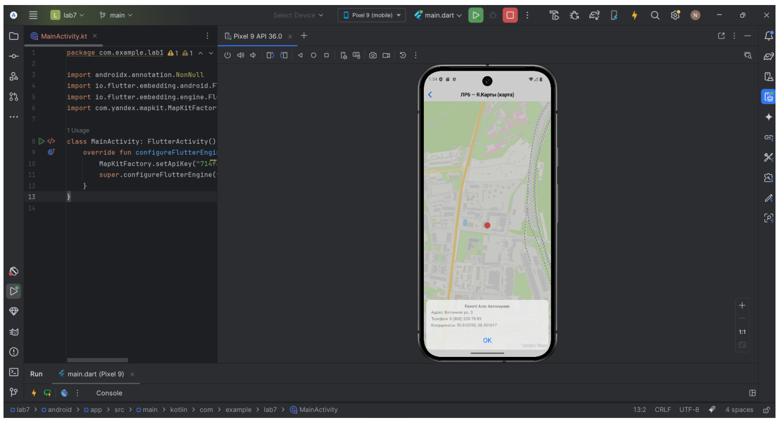
Исходный код программы представлен в листинге 1.

```
import 'dart:convert';
import 'dart:io' show SocketException;

import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;
import 'package:yandex_mapkit/yandex_mapkit.dart';

void main() {
  WidgetsFlutterBinding.ensureInitialized();
  runApp(const Lab6App());
}

///                              Cupertino ,
                                                          /
                                    .
class Lab6App extends StatelessWidget {
  const Lab6App({super.key});

  @override
  Widget build(BuildContext context) {
    return const CupertinoApp(
      debugShowCheckedModeBanner: false,
      title: ' 6                        .             ',
      home: Lab6YandexScreen(),
    );
  }
}
```

```

class Lab6YandexScreen extends StatefulWidget {
  const Lab6YandexScreen({super.key});
  @override
  State<Lab6YandexScreen> createState() => _Lab6YandexScreenState();
}

class _Lab6YandexScreenState extends State<Lab6YandexScreen> {
  static const String kDefaultEndpoint =
      'http://pstgu.yss.su/iu9/mobiledev/lab4_yandex_map/2023.php?x=var15';

  final _urlCtrl = TextEditingController(text: kDefaultEndpoint);

  bool _loading = false;
  String? _error;
  final List<_Org> _orgs = [];

  @override
  void dispose() {
    _urlCtrl.dispose();
    super.dispose();
  }

  Future<void> _load() async {
    final url = _urlCtrl.text.trim();
    if (url.isEmpty) return;

    setState(() {
      _loading = true;
      _error = null;
      _orgs.clear();
    });

    try {
      //                              ...
      final body = await _fetchBody(url);
      await _parseAndSet(body);
    } on SocketException catch (_) {
      //          DNS/host lookup
              https-              (                              )
      final proxy = _wrapWithProxy(url);
      try {
        final body = await _fetchBody(proxy);
      await _parseAndSet(body);
      _toast('                                                    .
                                                .');
    } catch (e2) {
      _error = '                                  : $e2';
    }
  } catch (e) {
    _error = '                                  : $e';
  } finally {
    if (mounted) setState(() => _loading = false);
  }
}

Future<String> _fetchBody(String url) async {
  final r = await http.get(Uri.parse(url)).timeout(const Duration
 (seconds: 20));
  if (r.statusCode != 200) {
    throw Exception('HTTP ${r.statusCode}');
  }
  return const Utf8Decoder(allowMalformed: true).convert(r.
 bodyBytes);
}

Future<void> _parseAndSet(String body) async {
  final jsonArrayString = _firstJsonArray(body);
  if (jsonArrayString == null) {
    throw Exception('                                       JSON-
                                ');
  }
  final rawList = jsonDecode(jsonArrayString);
  if (rawList is! List) {
    throw Exception('
                 ');
  }
  for (final raw in rawList) {
    if (raw is Map) {
      final org = _Org.tryParse(raw);
      if (org != null) _orgs.add(org);
    }
  }
  if (_orgs.isEmpty) {
    throw Exception('                                          ');
  }
  setState(() {}); //                    UI
}
```

```

  String _wrapWithProxy(String src) {
    if (src.startsWith('https://')) return src;
    return 'https://r.jina.ai/$src'; //            http-https
  }

  String? _firstJsonArray(String s) {
    final start = s.indexOf('[');
    if (start < 0) return null;
    var level = 0;
    for (var i = start; i < s.length; i++) {
      final ch = s[i];
      if (ch == '[') level++;
      if (ch == ']') {
        level--;
        if (level == 0) return s.substring(start, i + 1);
      }
    }
    return null;
  }

  void _openMap({int? focusIndex}) {
    if (_orgs.isEmpty) return;
    Navigator.of(context).push(
      CupertinoPageRoute(
        builder: (_) => Lab6MapScreen(
          orgs: List<_Org>.from(_orgs),
          focusIndex: focusIndex,
        ),
      ),
    );
  }

  void _toast(String msg) {
    //
    showCupertinoDialog(
      context: context,
      builder: (_) => CupertinoAlertDialog(
        content: Text(msg),
        actions: [
          CupertinoDialogAction(
            onPressed: () => Navigator.pop(context),
            child: const Text('OK'),
          )
        ],
      ),
    );
  }

  @override
  Widget build(BuildContext context) {
    return CupertinoPageScaffold(
      navigationBar: const CupertinoNavigationBar(
        middle: Text('   6                         .            '),
      ),
      child: SafeArea(
        child: ListView(
          padding: const EdgeInsets.all(16),
          children: [
            _SectionCard(
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  const Text('                                   ',
                      style: TextStyle(fontSize: 16, fontWeight:
   FontWeight.w600)),
                  const SizedBox(height: 8),
                  CupertinoTextField(
                    controller: _urlCtrl,
                    placeholder: 'http://.../2023.php?x=var15',
                    keyboardType: TextInputType.url,
                    padding:
                        const EdgeInsets.symmetric(horizontal: 12,
   vertical: 12),
                    onSubmitted: (_) => _load(),
                  ),
                  const SizedBox(height: 10),
                  Row(
                    children: [
                      Expanded(
                        child: CupertinoButton.filled(
                          onPressed: _loading ? null : _load,
                          child: Text(_loading ? '...' : '
                     '),
                        ),
                      ),
                      const SizedBox(width: 8),
                      Expanded(
                        child: CupertinoButton(
                          onPressed: _orgs.isEmpty ? null : () => _openMap(),
                          child: const Text('
                          '),
                        ),
                      ),
                    ],
                  ),
                  if (_error != null) ...[
                    const SizedBox(height: 8),
                    Text(_error!,
                        style:
                            const TextStyle(color: CupertinoColors.systemRed)),
                  ],
                ],
              ),
            ),
            if (_loading)
              const Center(child: CupertinoActivityIndicator())
            else if (_orgs.isNotEmpty)
              _SectionCard(
                child: Column(
                  crossAxisAlignment: CrossAxisAlignment.start,
                  children: [
                    const Text('            ',
                        style: TextStyle(
                            fontSize: 16, fontWeight: FontWeight.w600)),
                    const SizedBox(height: 8),
                    ...List.generate(_orgs.length, (i) {
                      final o = _orgs[i];
                      return Container(
                        decoration: BoxDecoration(
                          border: Border(
                            bottom: BorderSide(
                              color: CupertinoColors.separator
                                  .resolveFrom(context),
                              width: i == _orgs.length - 1 ? 0 : 0.5,
                            ),
                          ),
                        ),
                        child: CupertinoButton(
                          padding: const EdgeInsets.symmetric(
                              vertical: 10, horizontal: 0),
                          onPressed: () => _openMap(focusIndex: i),
                          child: Align(
                            alignment: Alignment.centerLeft,
                            child: Column(
                              crossAxisAlignment: CrossAxisAlignment.start,
                              children: [
                                Text(o.title,
                                    style: const TextStyle(
                                        fontWeight: FontWeight.w600)),
                                if ((o.address ?? '').isNotEmpty)
                                  Padding(
                                    padding: const EdgeInsets.only(top: 2),
                                    child: Text(
                                      o.address!,
                                      style: const TextStyle(
                                        color:
                                            CupertinoColors.secondaryLabel,
                                        fontSize: 13,
                                      ),
                                    ),
                                  ),
                              ],
                            ),
                          ),
                        ),
                      );
                    }),
                  ],
                ),
              ),
          ],
        ),
      ),
    );
  }
}

///
class Lab6MapScreen extends StatefulWidget {
  final List<_Org> orgs;
  final int? focusIndex; //                null

  const Lab6MapScreen({
    super.key,
    required this.orgs,
    this.focusIndex,
  });

  @override
  State<Lab6MapScreen> createState() => _Lab6MapScreenState();
}

class _Lab6MapScreenState extends State<Lab6MapScreen> {
  YandexMapController? _map;
  late final List<MapObject> _objects;

  @override
  void initState() {
    super.initState();
    _objects = List.generate(widget.orgs.length, (i) {
      final org = widget.orgs[i];
      return CircleMapObject(
        mapId: MapObjectId('org_$i'),
        circle: Circle(
          center: Point(latitude: org.lat, longitude: org.lon),
          radius: 14,
        ),
        fillColor: Colors.red.withOpacity(0.85),
        strokeColor: Colors.white,
        strokeWidth: 1.0,
        onTap: (_, __) => _showOrg(org),
      );
    });
  }

  Future<void> _onMapCreated(YandexMapController c) async {
    _map = c;
    await _map!.moveCamera(
      CameraUpdate.newCameraPosition(
        const CameraPosition(
          target: Point(latitude: 55.751244, longitude: 37.618423),
          zoom: 9.5,
        ),
      ),
      animation:
          const MapAnimation(type: MapAnimationType.smooth, duration: 0.3),
    );

    if (widget.orgs.isEmpty) return;

    if (widget.focusIndex == null) {
      await _fitAll();
    } else {
      final org =
          widget.orgs[widget.focusIndex!.clamp(0, widget.orgs.length - 1)];
      await _focusOn(org);
      _showOrg(org);
    }
  }

  Future<void> _fitAll() async {
    if (_map == null || widget.orgs.isEmpty) return;

    final lats = widget.orgs.map((e) => e.lat);
    final lons = widget.orgs.map((e) => e.lon);
    final minLat = lats.reduce((a, b) => a < b ? a : b);
    final maxLat = lats.reduce((a, b) => a > b ? a : b);
    final minLon = lons.reduce((a, b) => a < b ? a : b);
    final maxLon = lons.reduce((a, b) => a > b ? a : b);

    final center = Point(
      latitude: (minLat + maxLat) / 2,
      longitude: (minLon + maxLon) / 2,
    );

    final dLat = (maxLat - minLat).abs();
    final dLon = (maxLon - minLon).abs();
    final spread = (dLat > dLon ? dLat : dLon);

    double zoom;
    if (spread < 0.005) zoom = 16;
    else if (spread < 0.02) zoom = 14;
    else if (spread < 0.1) zoom = 12;
    else if (spread < 0.5) zoom = 10;
    else zoom = 8;

    await _map!.moveCamera(
      CameraUpdate.newCameraPosition(
        CameraPosition(target: center, zoom: zoom),
      ),
      animation:
          const MapAnimation(type: MapAnimationType.smooth, duration: 0.7),
    );
  }

  Future<void> _focusOn(_Org org) async {
    if (_map == null) return;
    await _map!.moveCamera(
      CameraUpdate.newCameraPosition(
        CameraPosition(
          target: Point(latitude: org.lat, longitude: org.lon),
          zoom: 16,
        ),
      ),
      animation:
          const MapAnimation(type: MapAnimationType.smooth, duration: 0.6),
    );
  }

  void _showOrg(_Org org) {
    showCupertinoModalPopup(
      context: context,
      builder: (_) => CupertinoActionSheet(
        title: Text(org.title, textAlign: TextAlign.center),
        message: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            if ((org.address ?? '').isNotEmpty) Text('          : ${org.address}'),
            if ((org.info ?? '').isNotEmpty)
              Padding(
                padding: const EdgeInsets.only(top: 6),
                child: Text(org.info!),
              ),
            Padding(
              padding: const EdgeInsets.only(top: 6),
              child: Text(
                '                    : ${org.lat.toStringAsFixed(6)}, ${org.lon.toStringAsFixed(6)}',
                style: const TextStyle(
                  color: CupertinoColors.secondaryLabel,
                ),
              ),
            ),
          ],
        ),
        actions: [
          CupertinoActionSheetAction(
            onPressed: () => Navigator.pop(context),
            child: const Text('OK'),
          ),
        ],
      ),
    );
  }

  @override
  Widget build(BuildContext context) {
    return CupertinoPageScaffold(
      navigationBar: const CupertinoNavigationBar(
        middle: Text(' 6        .          (          )'),
      ),
      child: SafeArea(
        child: YandexMap(
          onMapCreated: _onMapCreated,
          mapObjects: _objects,
        ),
      ),
    );
  }
}

//                       /                           JSON

class _Org {
  final String title;
  final String? address;
  final String? info; //                                ,

  final double lat;
  final double lon;

  _Org({
    required this.title,
    this.address,
    this.info,
    required this.lat,
    required this.lon,
  });

  static double? _toD(dynamic v) {
    if (v == null) return null;
    if (v is num) return v.toDouble();
    if (v is String) return double.tryParse(v.replaceAll(',', '.'))
      ;
    return null;
  }

  ///                                    JSON             :
  /// [{"name": "...", "gps": "55.78, 38.43", "address": "...", "
    tel": "..."}]
  static _Org? tryParse(Map raw) {
    final title = (raw['name'] ?? raw['title'] ?? '').toString().
    trim();
    final addr = raw['address']?.toString();

    final tel = raw['tel']?.toString();
    final info = (tel != null && tel.isNotEmpty) ? '                :
      $tel' : null;

    double? lat, lon;

    // gps: "55.780359, 38.434721"
    final gps = raw['gps'];
    if (gps != null) {
      final nums = RegExp(r'-?\d+(?:[.,]\d+)?')
          .allMatches(gps.toString())
          .map((m) => m.group(0)!.replaceAll(',', '.'))
          .toList();
      if (nums.length >= 2) {
        lat = double.tryParse(nums[0]);
        lon = double.tryParse(nums[1]);
      }
    }

    lat ??= _toD(raw['lat'] ?? raw['latitude'] ?? raw['y']);
    lon ??= _toD(raw['lon'] ?? raw['lng'] ?? raw['longitude'] ??
    raw['x']);

    if (lat == null || lon == null) return null;

    return _Org(
      title: title.isEmpty ? '              ' : title,
      address: addr,
      info: info,
      lat: lat,
      lon: lon,
    );
  }
}

//                                                 -
   iOS

class _SectionCard extends StatelessWidget {
  final Widget child;
  const _SectionCard({required this.child});

  @override
  Widget build(BuildContext context) {
    return Container(
      margin: const EdgeInsets.only(bottom: 16),
      padding: const EdgeInsets.all(12),
      decoration: BoxDecoration(
        color: CupertinoColors.systemBackground.resolveFrom(context
    ),
        borderRadius: BorderRadius.circular(12),
        boxShadow: [
          BoxShadow(
            blurRadius: 6,
            color: Colors.black.withOpacity(0.06),
            offset: const Offset(0, 2),
          ),
        ],
        border: Border.all(
          color: CupertinoColors.separator.resolveFrom(context),
          width: 0.5,
        ),
      ),
      child: child,
    );
    }
}
```

Результат запуска представлен на рисунке 1.



```
package com.example.lab1

import androidx.annotation.NonNull
import io.flutter.embedding.android.F
import io.flutter.embedding.engine.Fl
import com.yandex.mapkit.MapKitFactor

1 Usage
class MainActivity: FlutterActivity()
    override fun configureFlutterEngi
        MapKitFactory.setApiKey("714f
        super.configureFlutterEngine(
    }
}
```

ЛР6 — Я.Карты (карта)

Favorit Auto Автосервис
Адрес: Бетонная ул., 3
Телефон: 8 (800) 250-78-89
Координаты: 55.842593, 38.431017

Рис. 1 — Результат

.