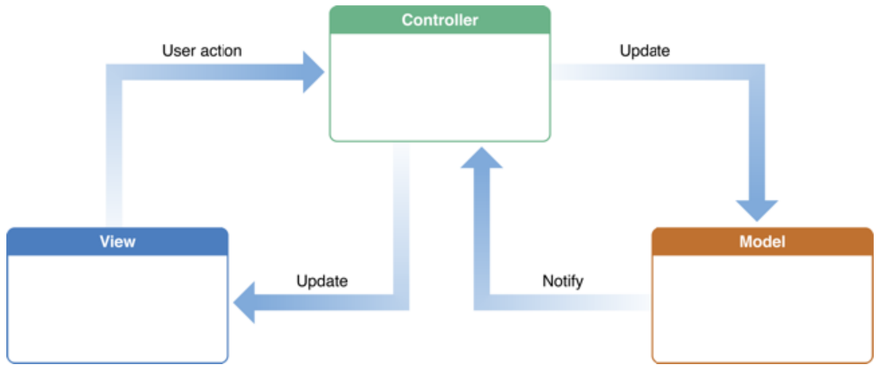
# Application Coordinator в iOS приложениях

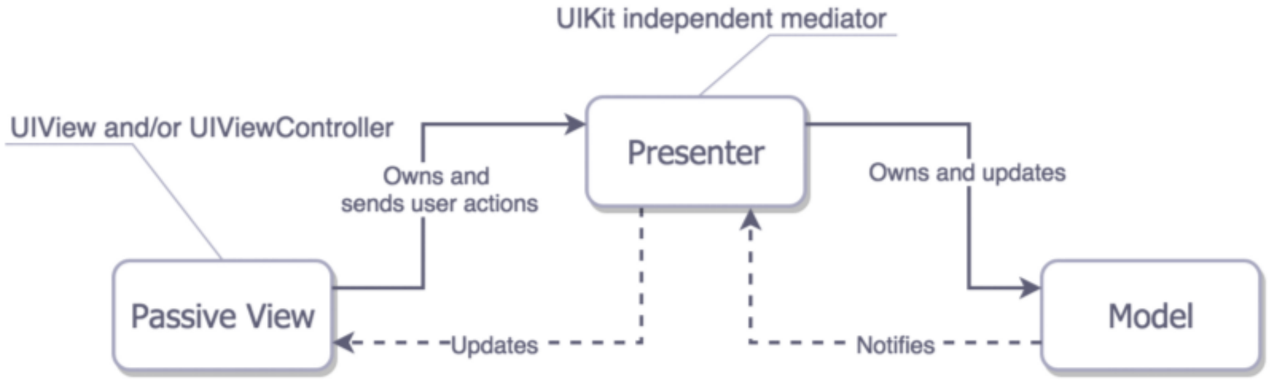
Павел Гуров
pegurov@avito.ru

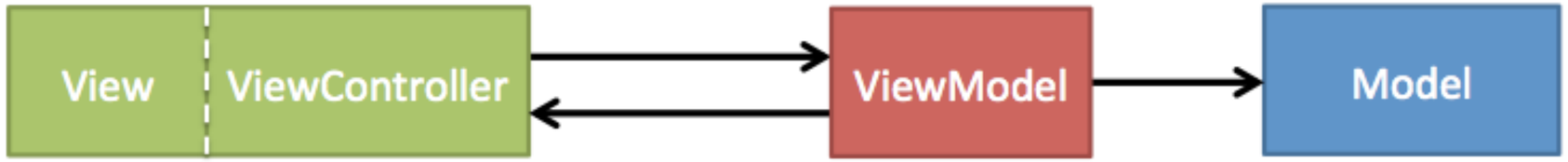
Avito

Avito
Apps Conf
App's 2017

# MVx


Controller
User action
Update
View
Update
Notify
Model


UIKit independent mediator
UIView and/or UIViewController
Presenter
Owns and
sends user actions
Owns and updates
Passive View
Updates
Notifies
Model


View
ViewController
ViewModel
Model

# UIStoryboard + segues

UIStoryboard + segues

UINavigationController

Avito



Apps Conf 2017

UIStoryboard + segues

UINavigationController

present(_:animated:completion:)

UIStoryboard + segues

UINavigationController

present(_:animated:completion:)

# UIStoryboard + segues

```
self.performSegue(withIdentifier: "", sender: self)
```

# UINavigationController

```
self.navigationController
```

# present(_:animated:completion:)

```
self.present(next, animated: true, completion: nil)
```

Avito



Apps Conf

```
/*
// MARK: - Navigation

// In a storyboard-based application,
you will often want to do a little
preparation before navigation

override func prepare(
    for segue: UIStoryboardSegue,
    sender: Any?) {

    // Get the new view controller using
    // segue.destinationViewController.
    // Pass the selected object to the
    // new view controller.
}
*/
```

Avito



Apps Conf

Apps 2017

```
/*
// MARK: - Navigation

// In a storyboard-based application,
you will often want to do a little
preparation before navigation

override func prepare(
    for segue: UIStoryboardSegue,
    sender: Any?) {

    // Get the new view controller using
    // segue.destinationViewController.
    // Pass the selected object to the
    // new view controller.
}
*/
```

Avito



Apps Conf

Apps 2017

```
/*
// MARK: - Navigation

// In a storyboard-based application,
you will often want to do a little
preparation before navigation

override func prepare(
    for segue: UIStoryboardSegue,
    sender: Any?) {

    // Get the new view controller using
    // segue.destinationViewController.
    // Pass the selected object to the
    // new view controller.
}
*/
```

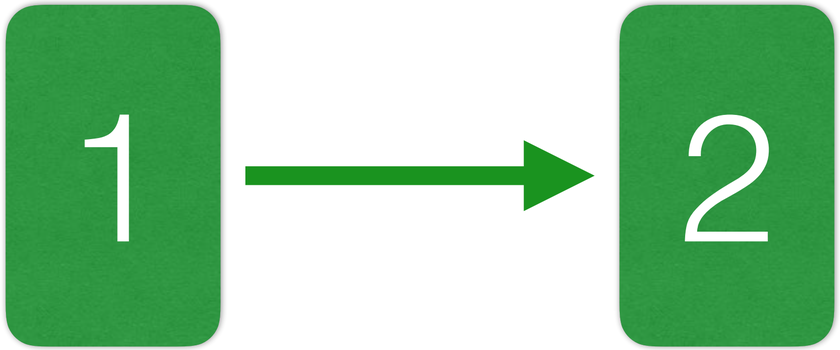
Avito



Apps Conf

App's 2017

# Жесткая связанность

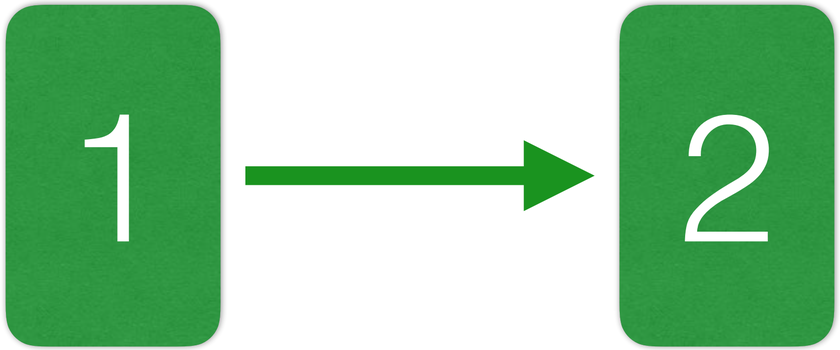



Avito



Apps Conf
App's 2017

# Жесткая связанность

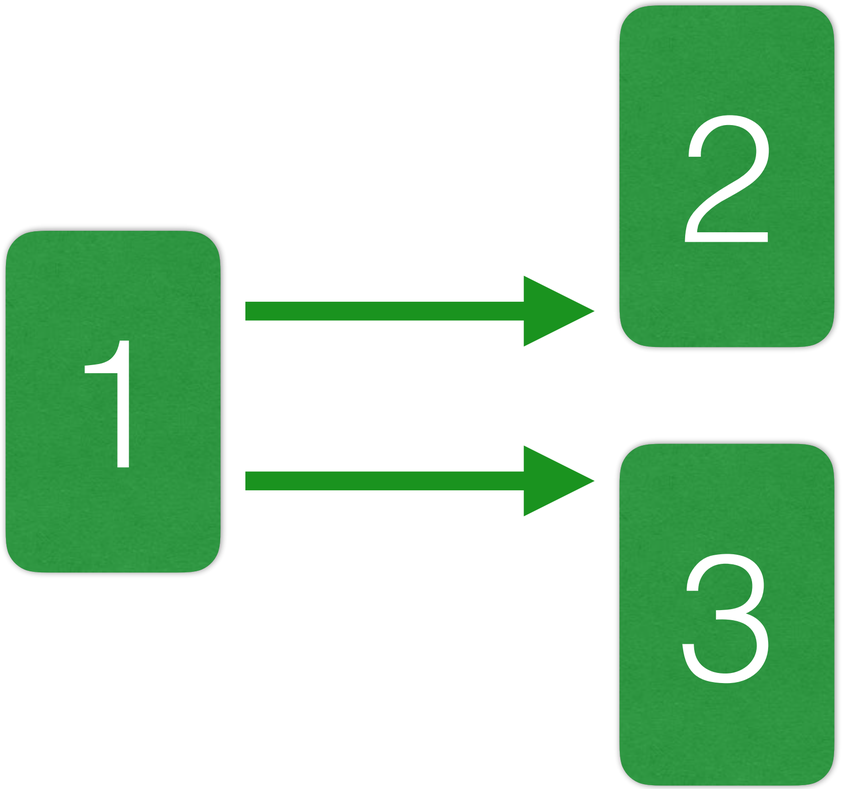



Avito



Apps Conf
App's 2017

# Жесткая связанность

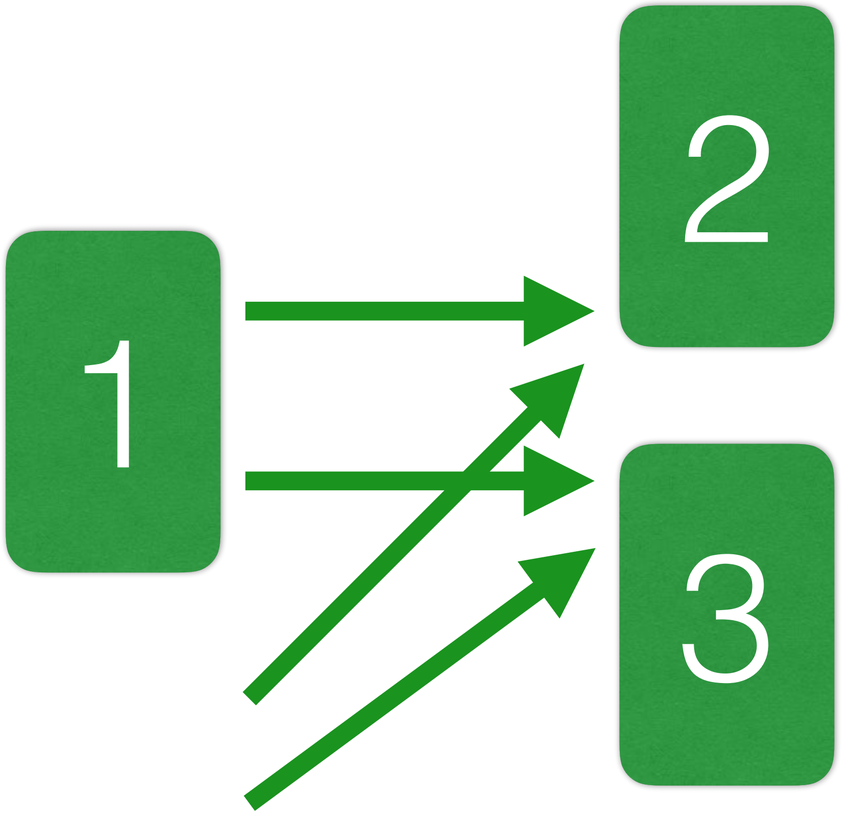



Avito



Apps Conf 2017

# Жесткая связанность

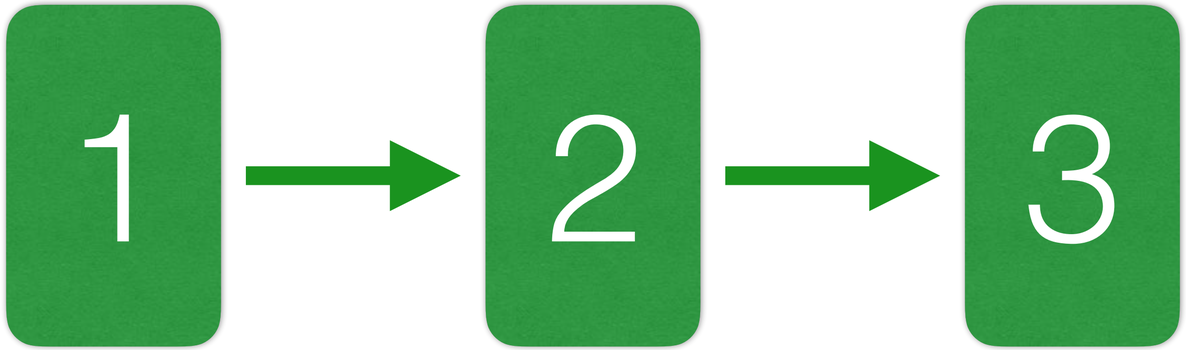



Avito



Apps Conf 2017

# Порядок

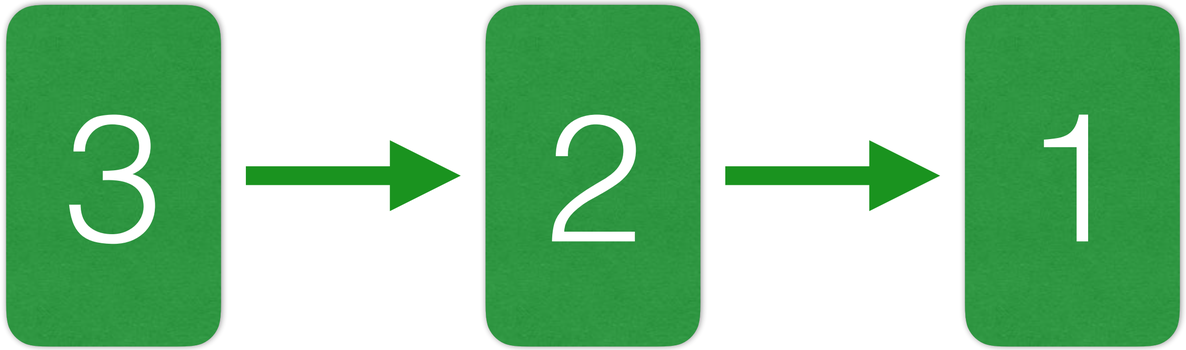



Avito



Apps Conf
App's 2017

# Порядок

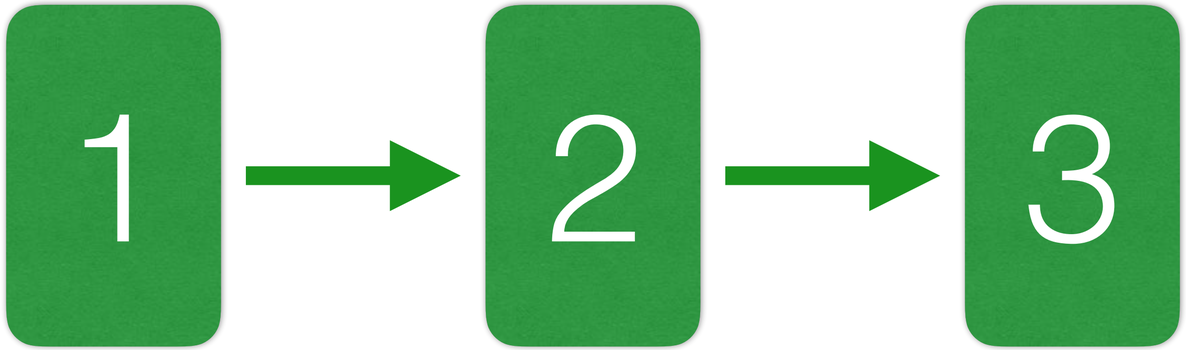



Avito



Apps Conf
App's 2017

# Передача данных

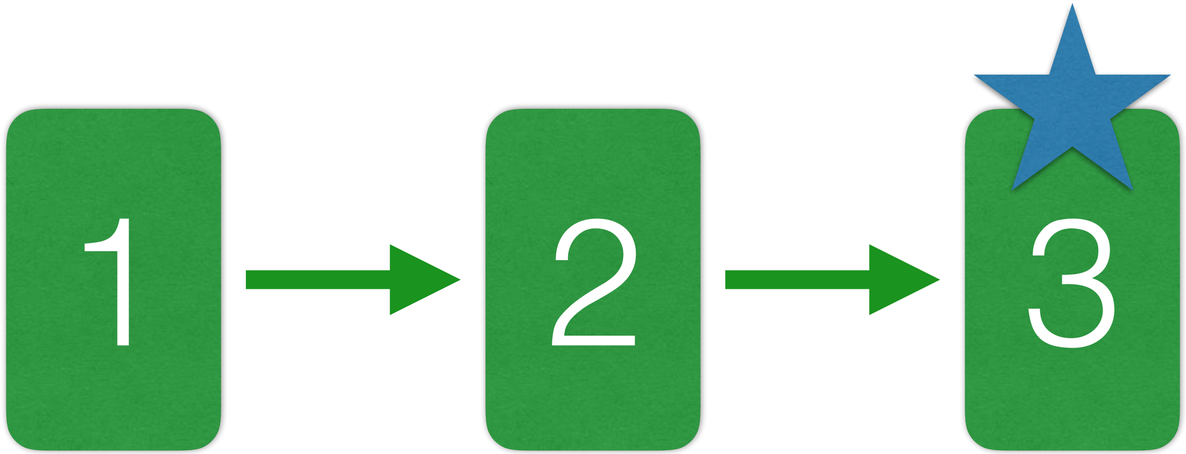



Avito

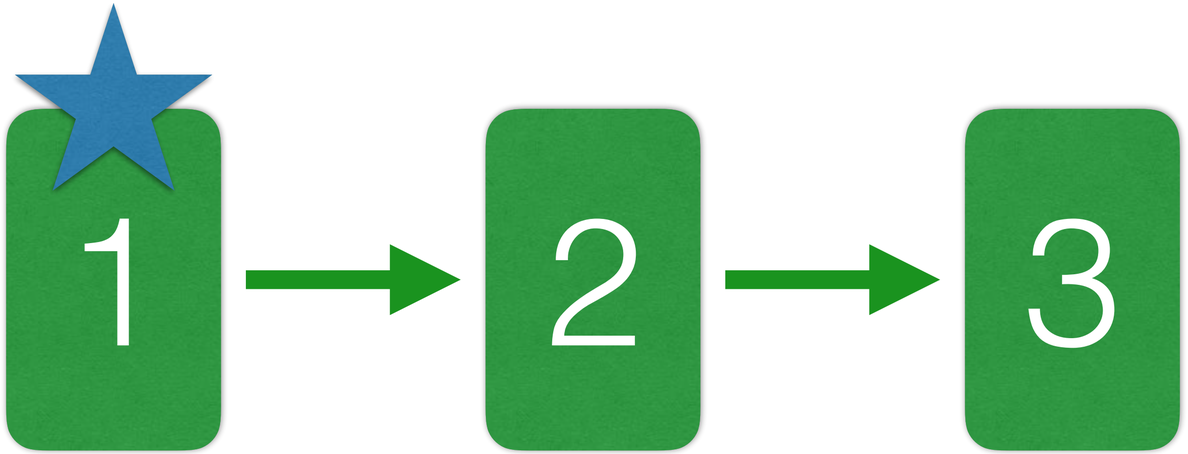


Apps Conf 2017

# Передача данных

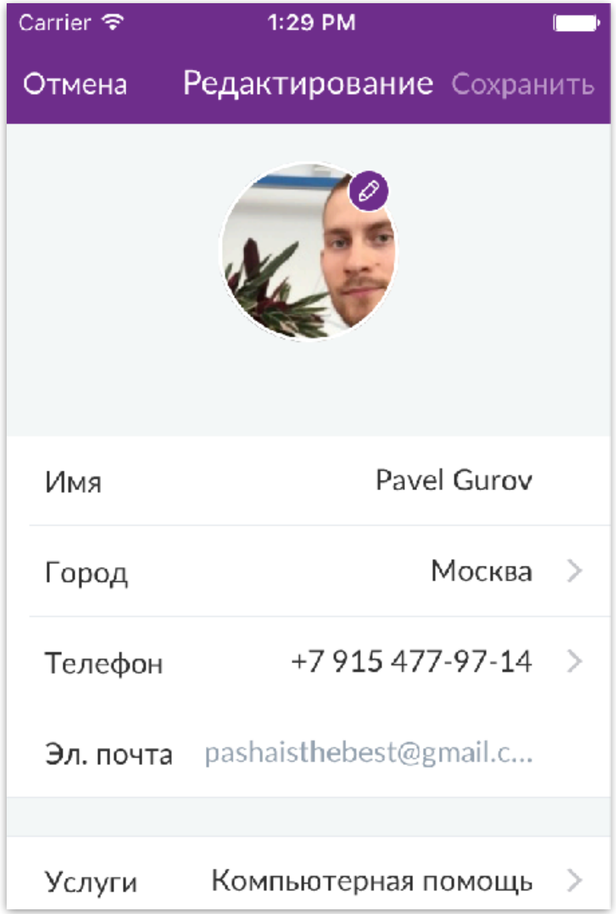


# Передача данных

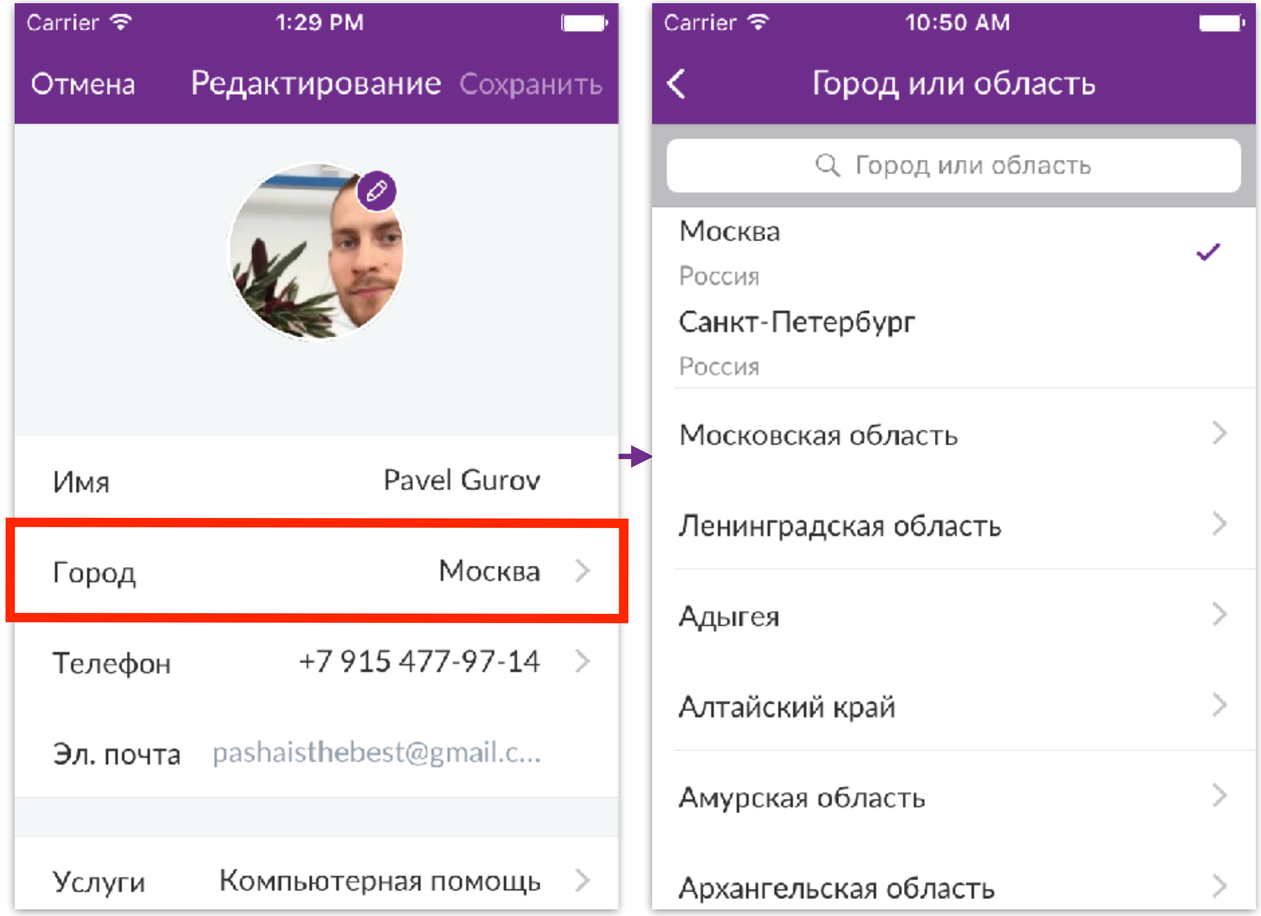


# Выбор города

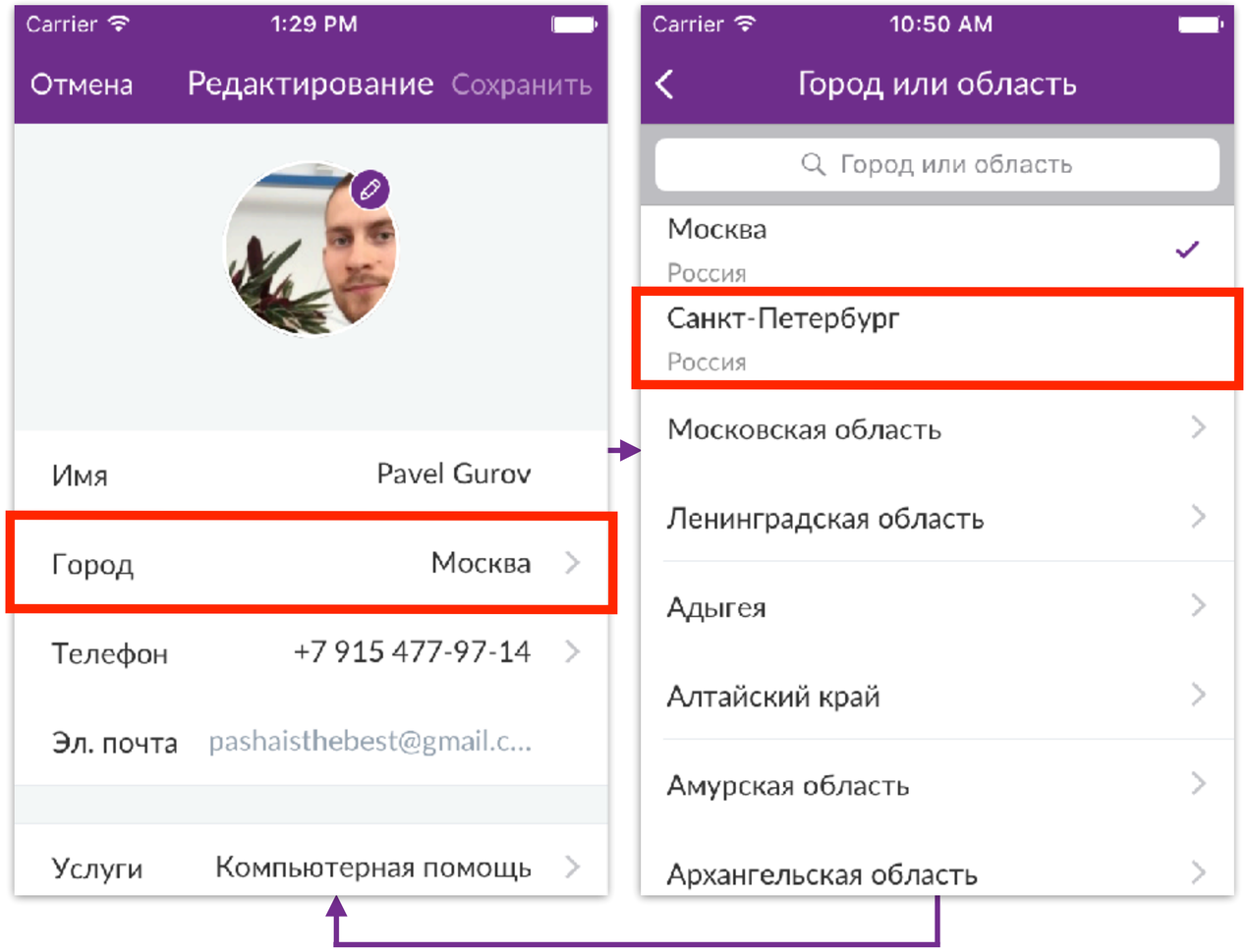
# Выбор города



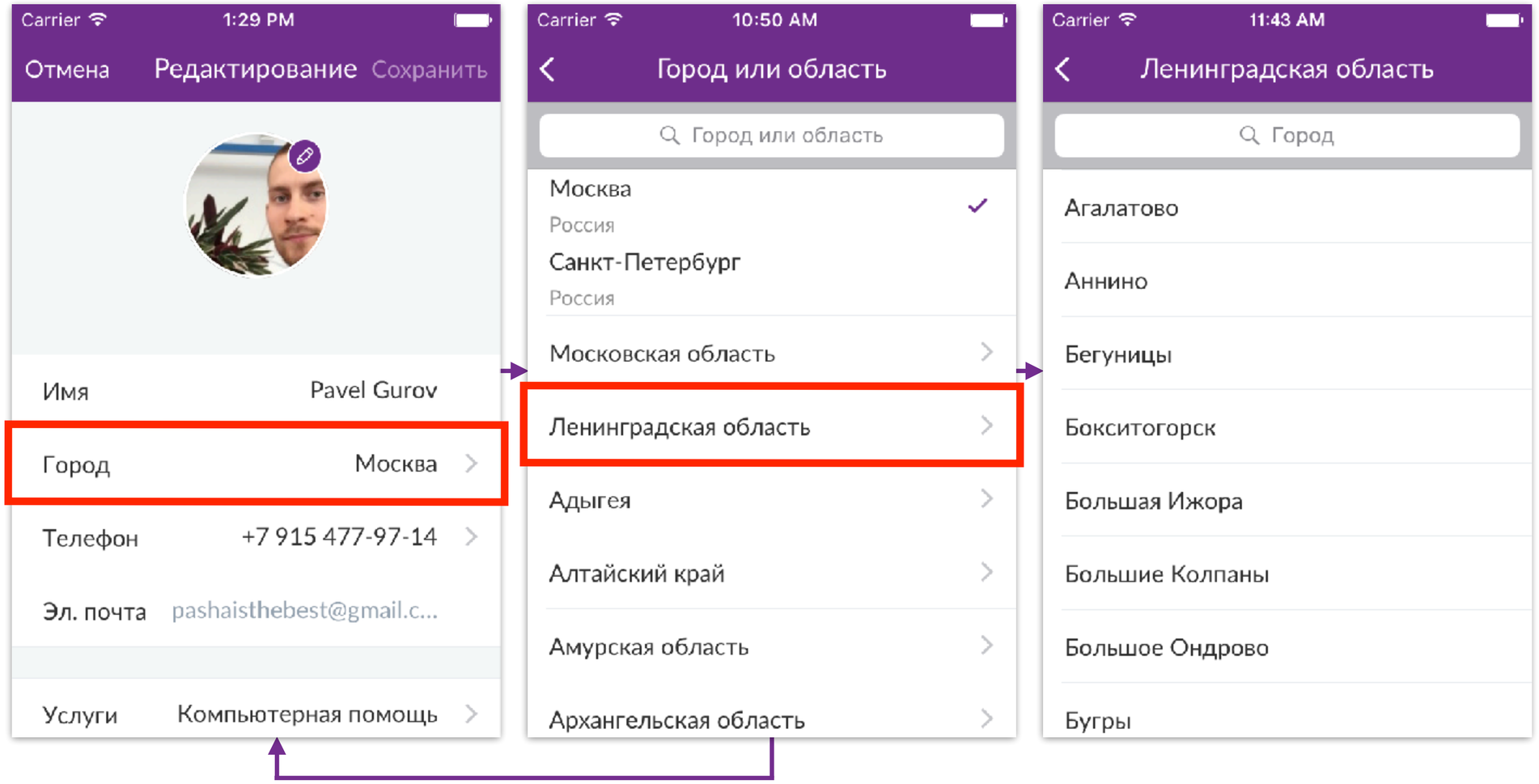
# Выбор города



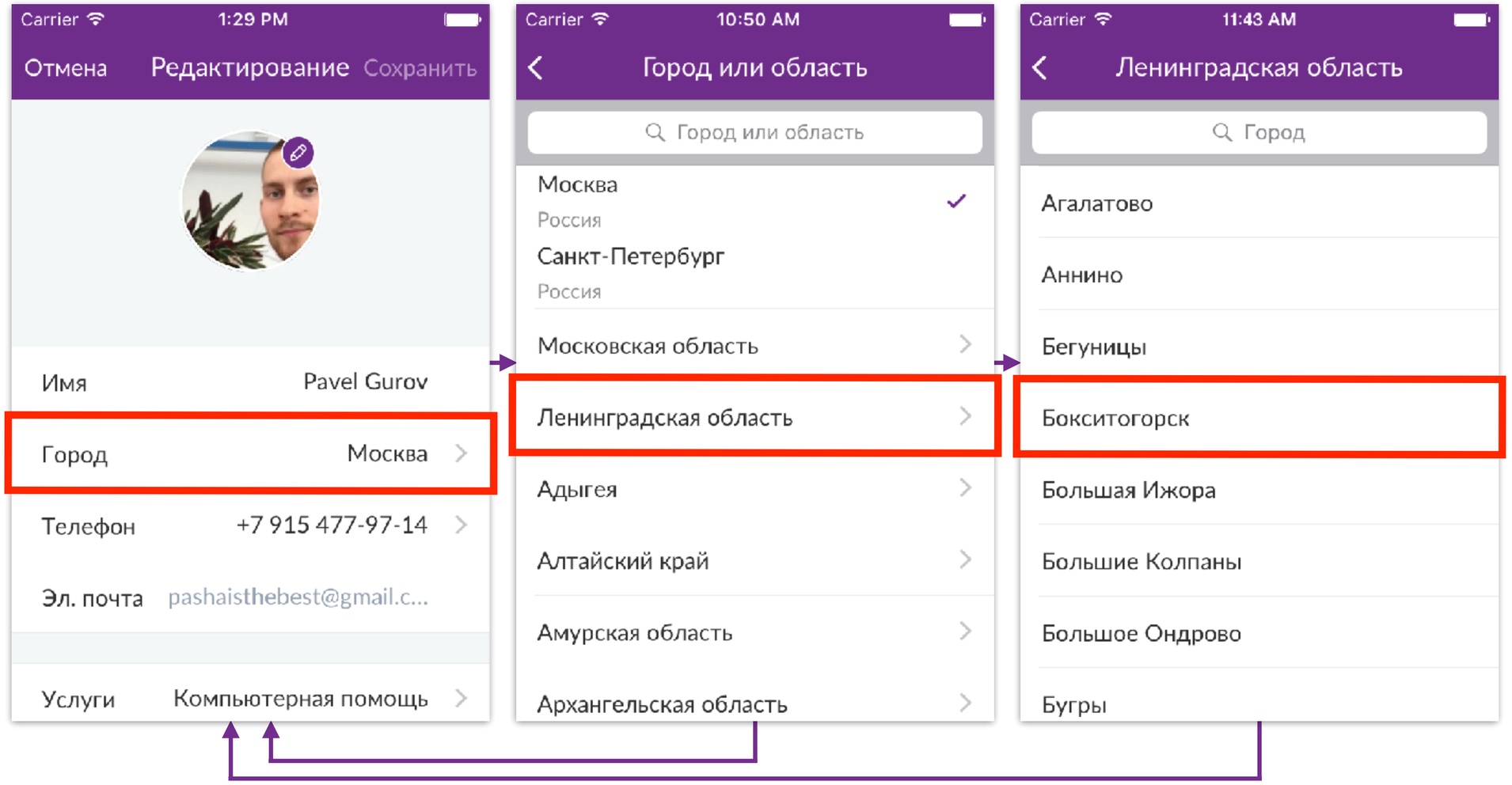
# Выбор города

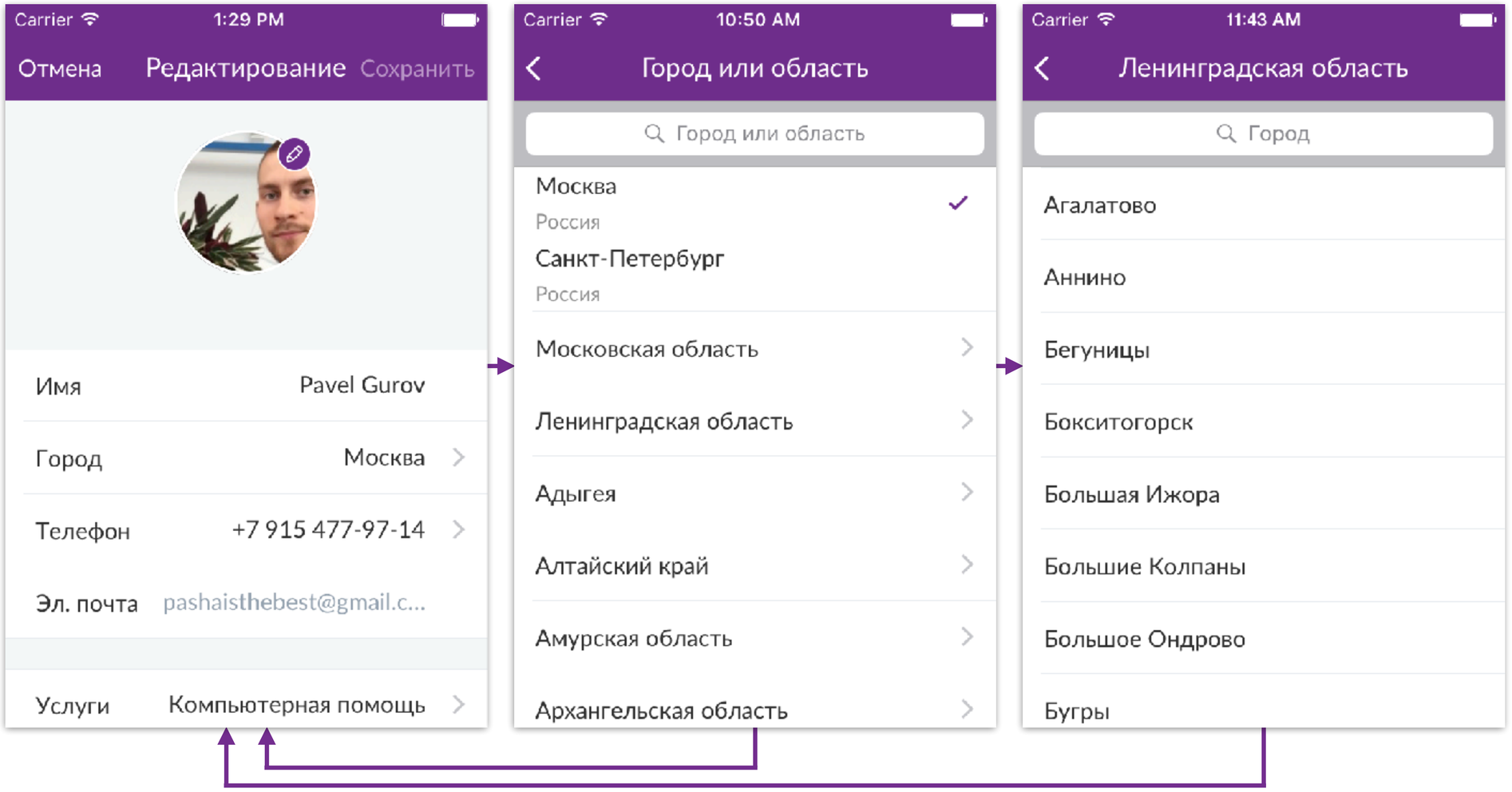


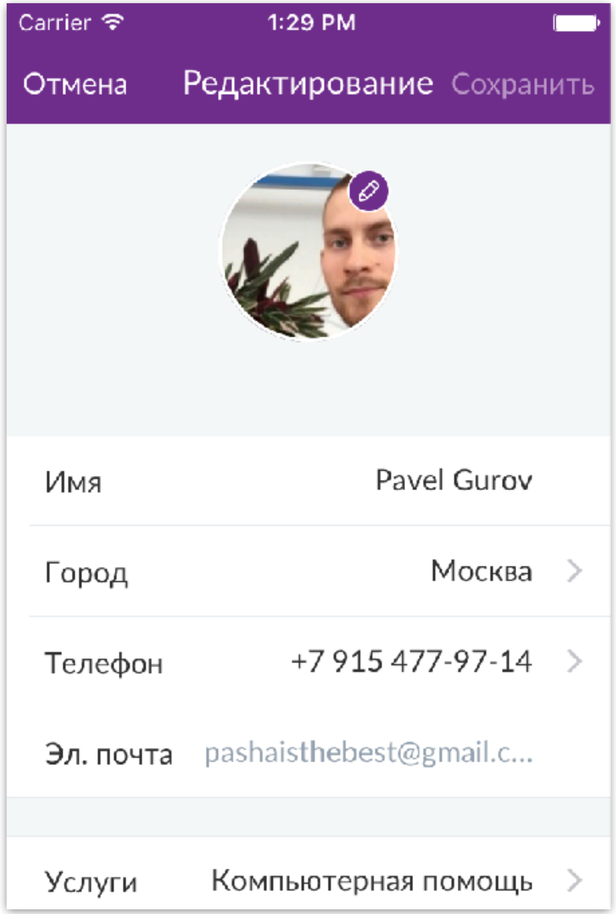
# Выбор города

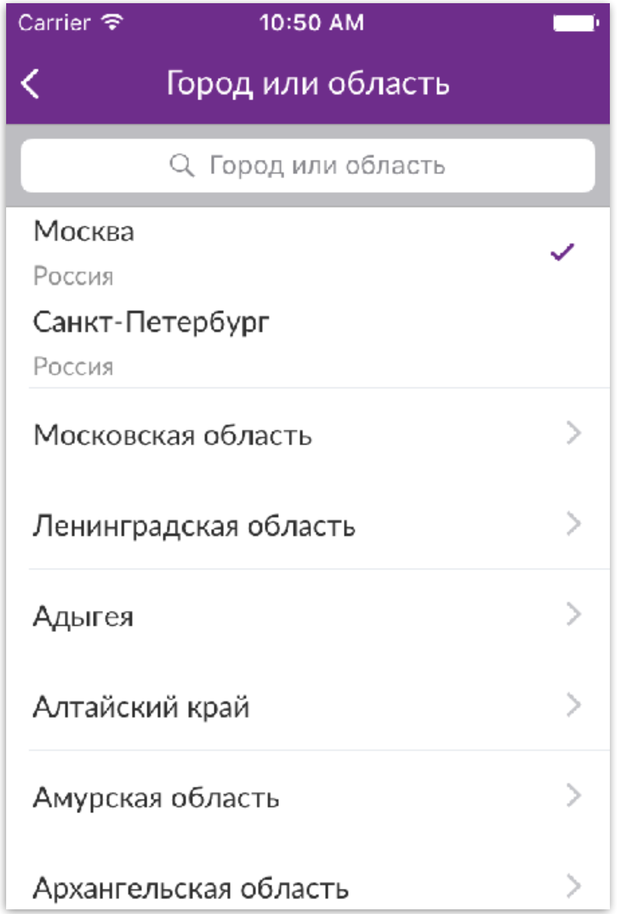


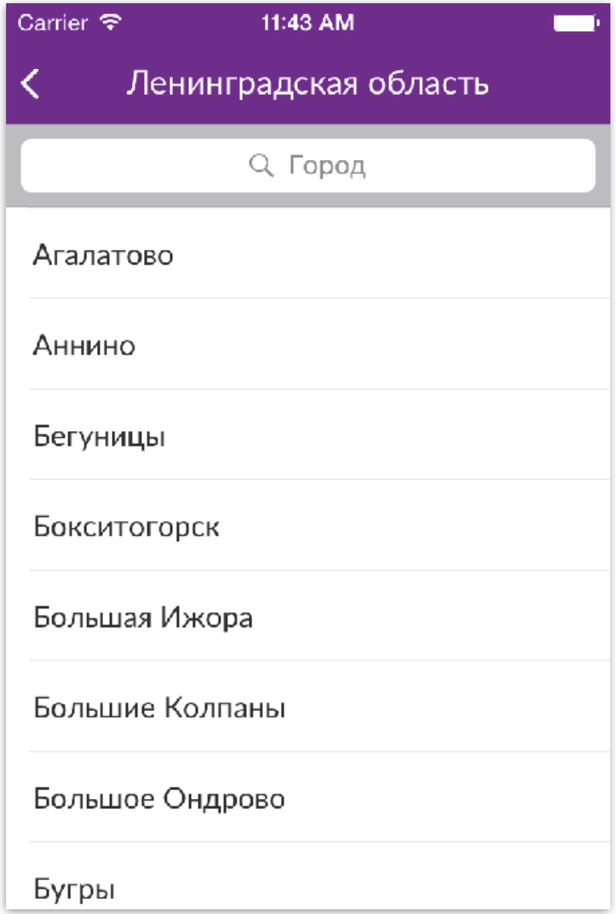
# Выбор города




Carrier 1:29 PM
Отмена
Редактирование
Сохранить
Имя
Pavel Gurov
Город
Москва
Телефон
+7 915 477-97-14
Эл. почта
pashaisthebest@gmail.c...
Услуги
Компьютерная помощь
Carrier 10:50 AM
Город или область
Город или область
Москва
Россия
Санкт-Петербург
Россия
Московская область
Ленинградская область
Адыгея
Алтайский край
Амурская область
Архангельская область
Carrier 11:43 AM
Ленинградская область
Город
Агалатово
Аннино
Бегуницы
Бокситогорск
Большая Ижора
Большие Колпаны
Большое Ондрово
Бугры




Avito



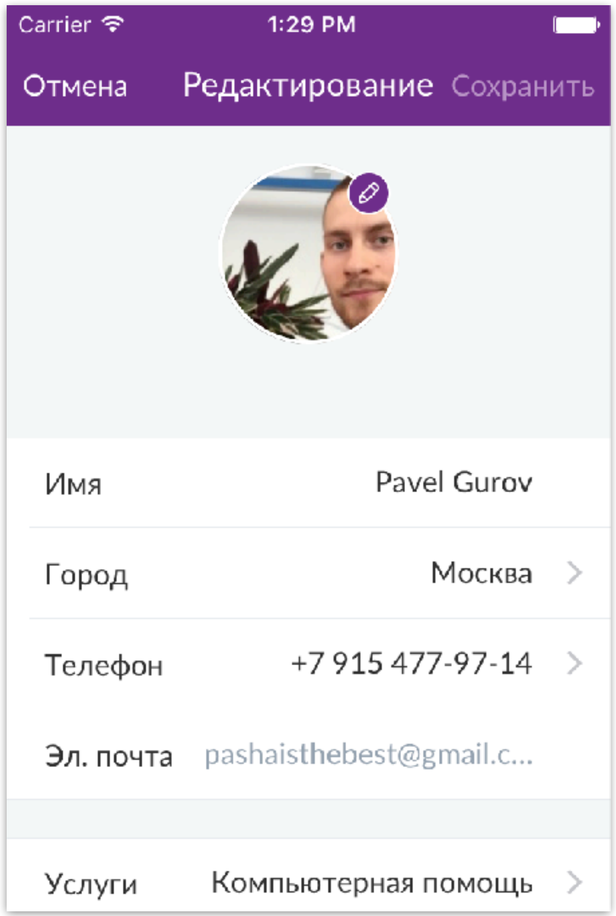
Apps Conf 2017

```
self.navigationController
self.tabBarController
```

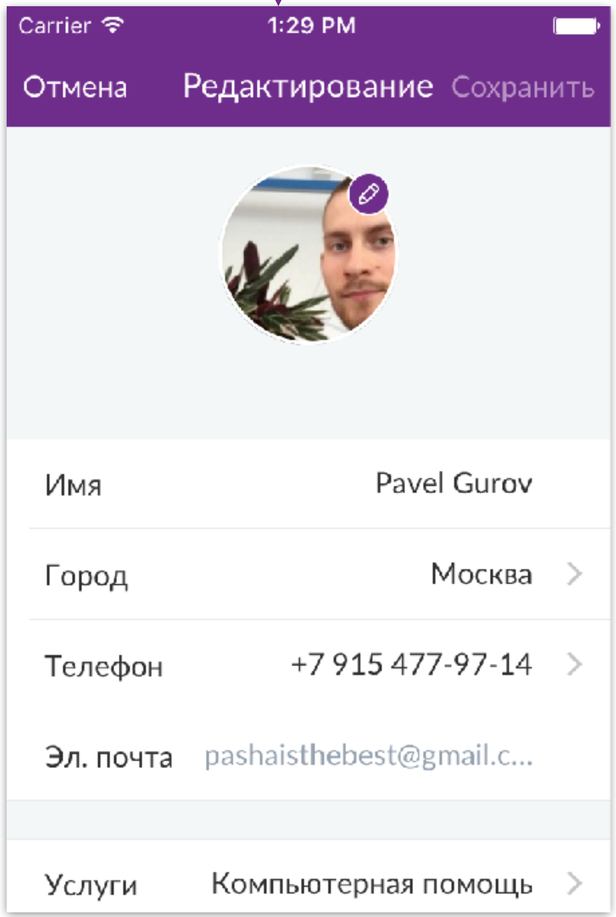
```
performSegue(
    withIdentifier: "next",
    sender: self
)
prepare(
    for segue: UIStoryboardSegue,
    sender: Any?
)
```

```
let next = CitiesViewController()
next.city = City(name: "Moscow")
```

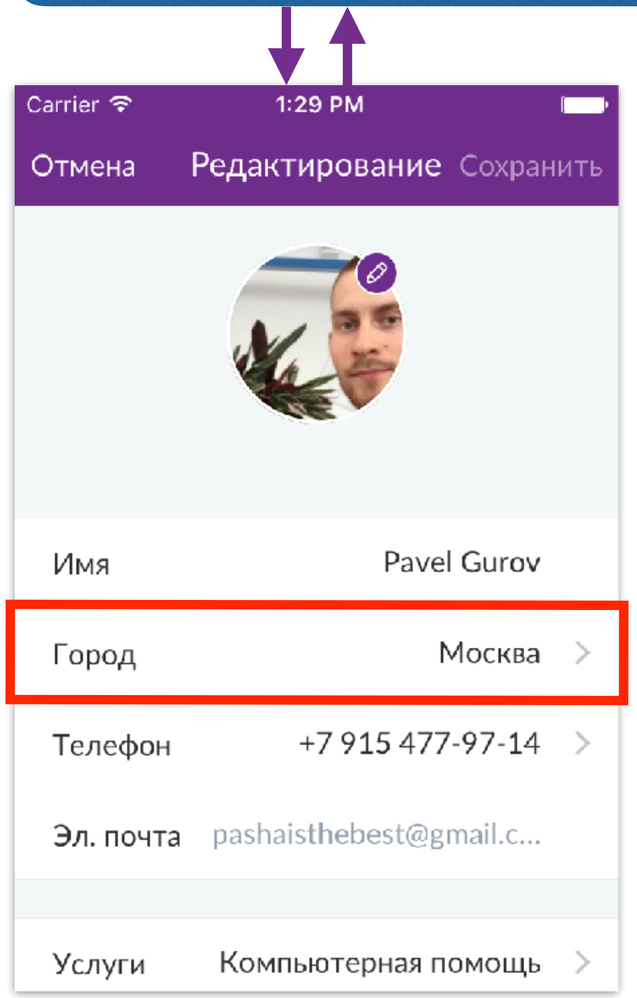
# Coordinator

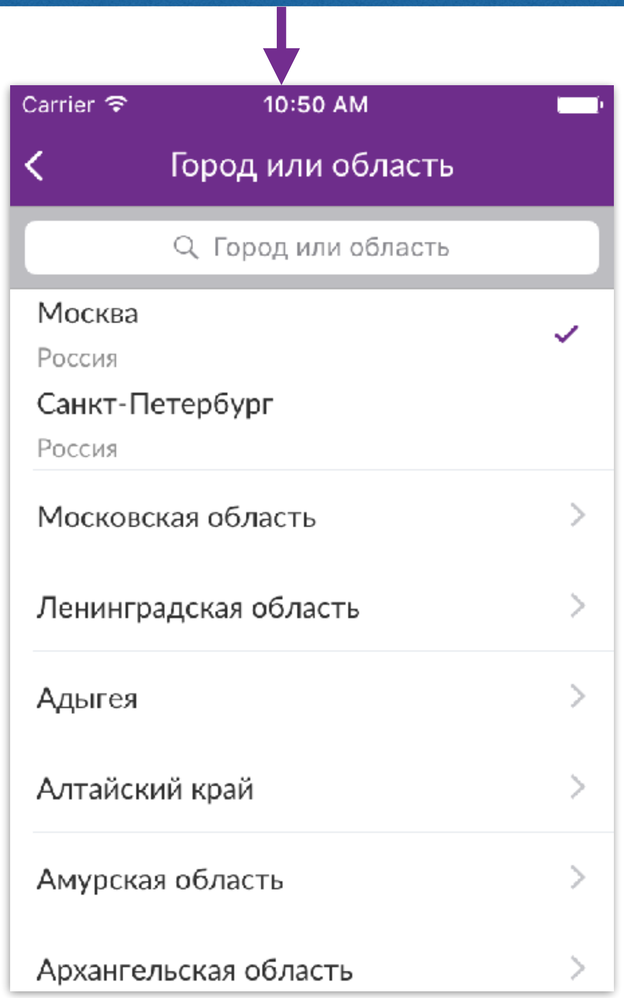


# Coordinator



# Coordinator



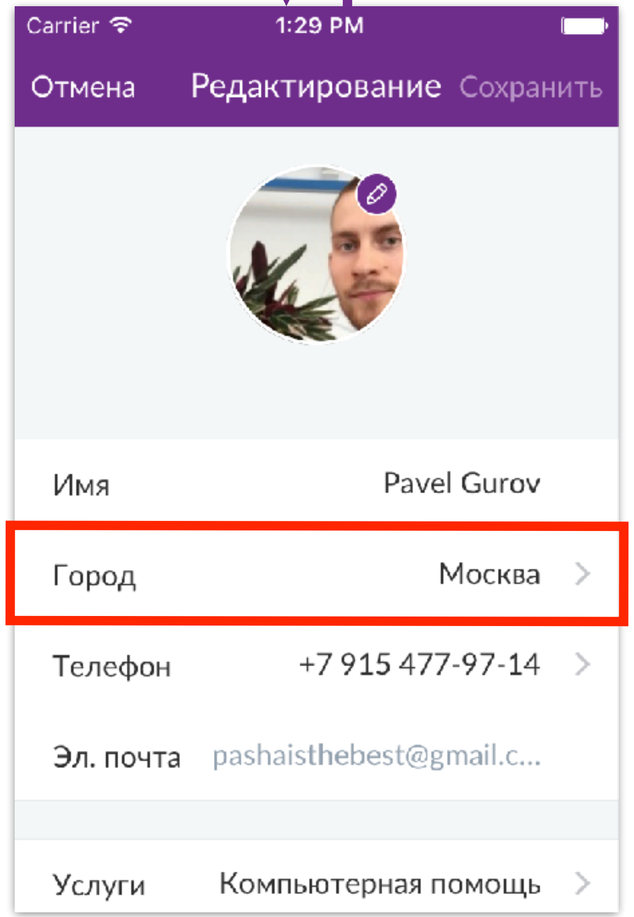



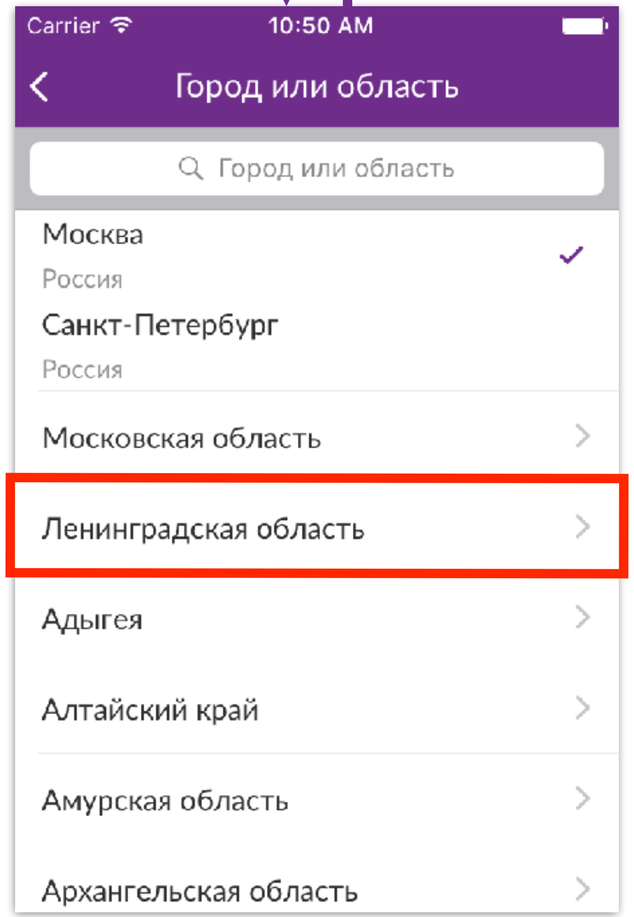

Avito

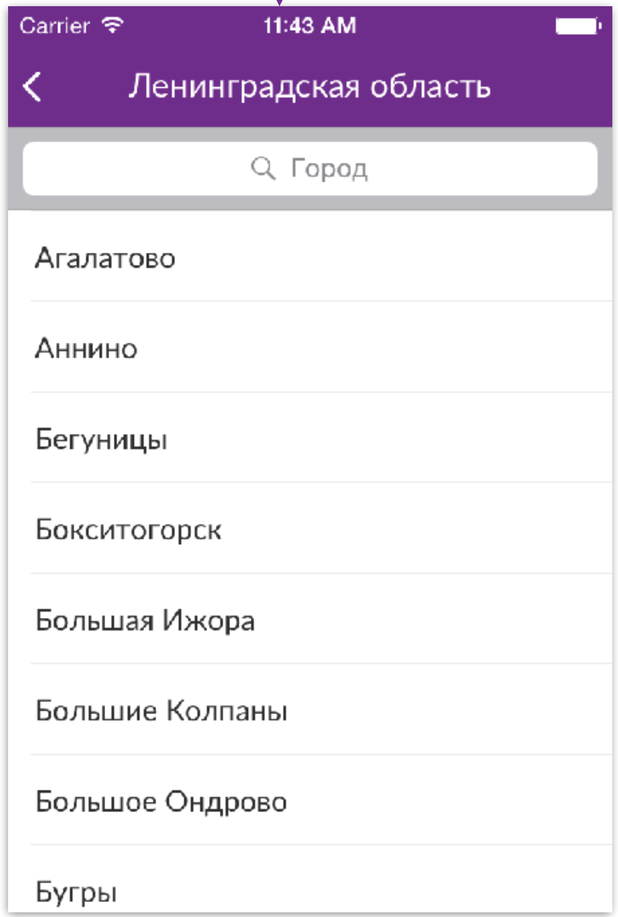


Apps Conf 2017

# Coordinator



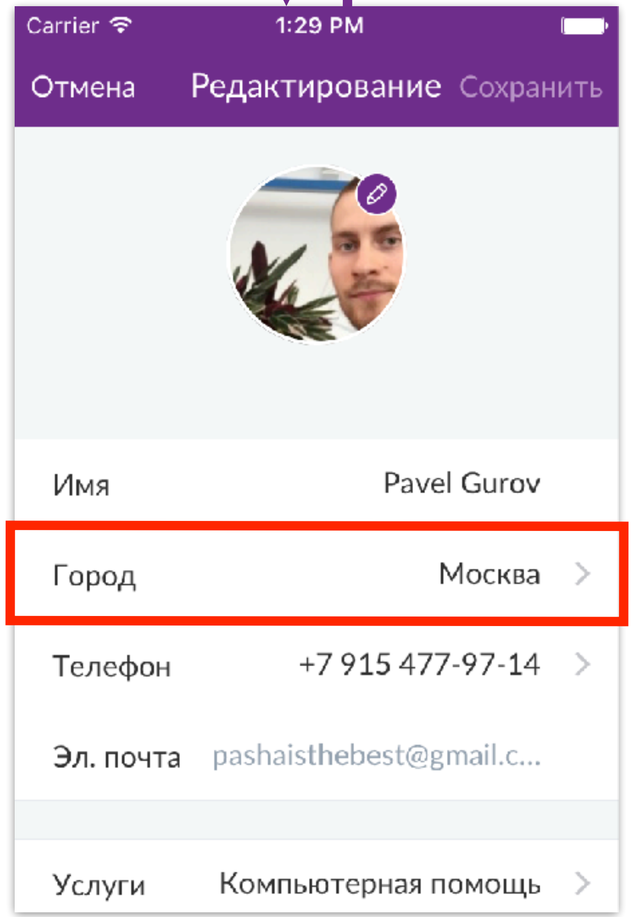



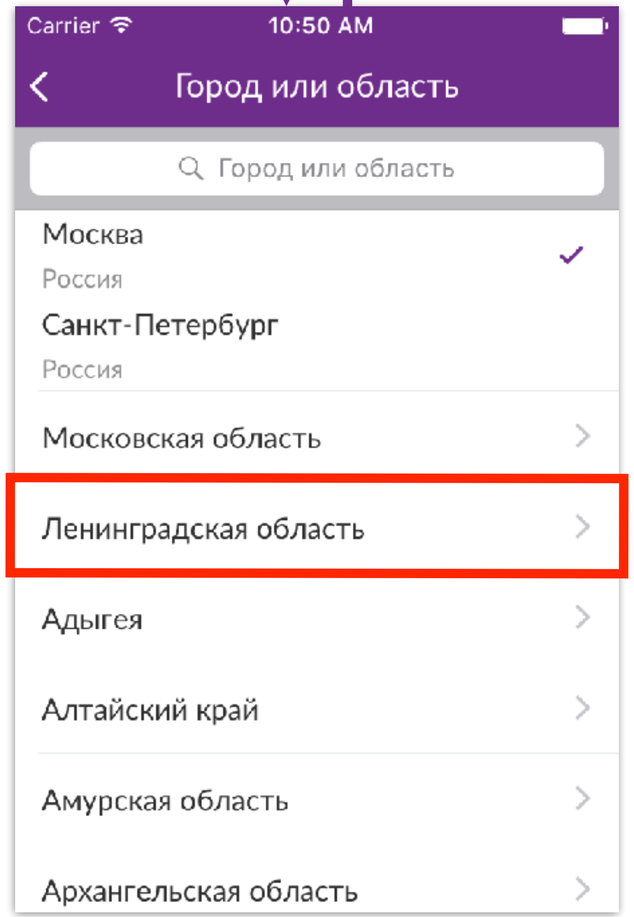

Avito

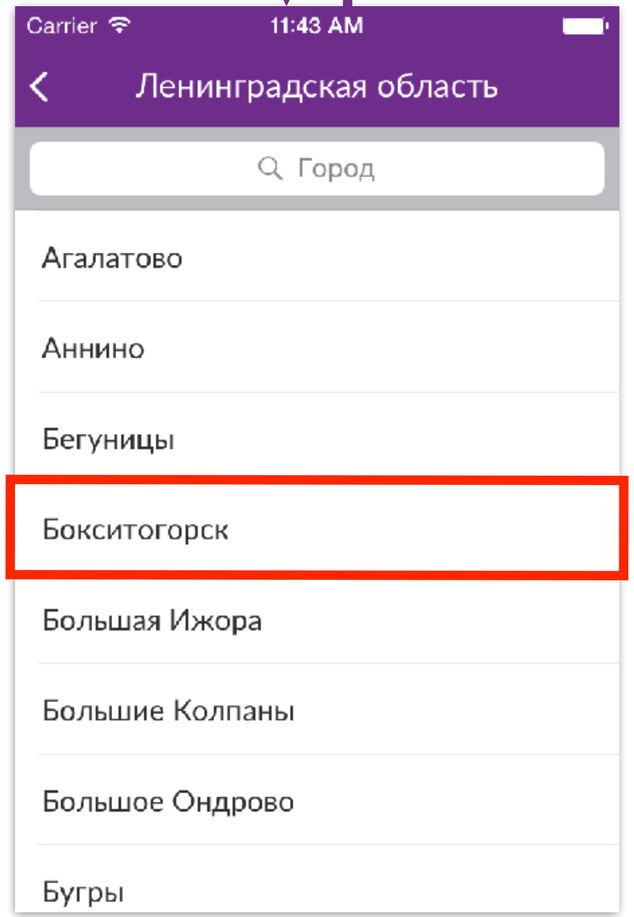


Apps Conf

# Coordinator



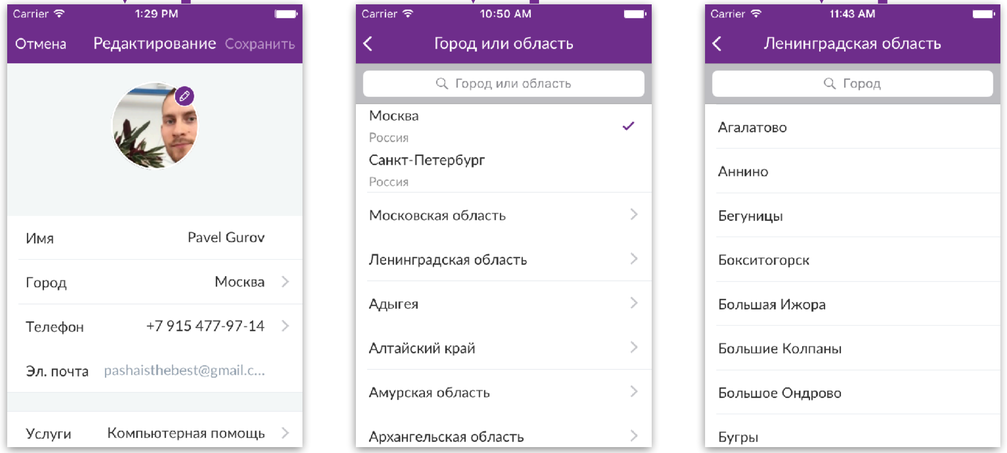




Avito



Apps Conf

# Coordinator



Data

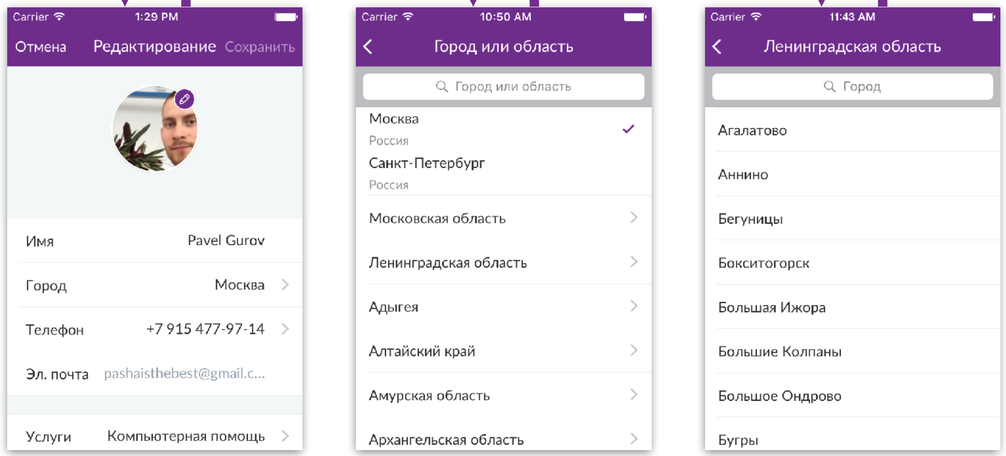
Database

Storage

Logic

Coordinator

Presentation




Avito



Apps Conf 2017

# Demo

# Inputs and outputs

```
class UserEditViewController: UIViewController {

    // MARK: - Input -
    var user: User? { didSet { updateView() } }

    // MARK: - Output -
    var onSelectCity: (() -> Void)?
}


class CitiesViewController: UITableViewController {

    // MARK: - Output -
    var onCitySelected: ((City) -> Void)?
}
```

Avito



Apps Conf

# Inputs and outputs

```
class UserEditViewController: UIViewController {

    // MARK: - Input -
    var user: User? { didSet { updateView() } }

    // MARK: - Output -
    var onSelectCity: (() -> Void)?
}


class CitiesViewController: UITableViewController {

    // MARK: - Output -
    var onCitySelected: ((City) -> Void)?
}
```

Avito



Apps Conf
App's 2017

# Inputs and outputs

```
class UserEditViewController: UIViewController {

    // MARK: - Input -
    var user: User? { didSet { updateView() } }

    // MARK: - Output -
    var onSelectCity: (() -> Void)?
}

class CitiesViewController: UITableViewController {

    // MARK: - Output -
    var onCitySelected: ((City) -> Void)?
}
```

Avito

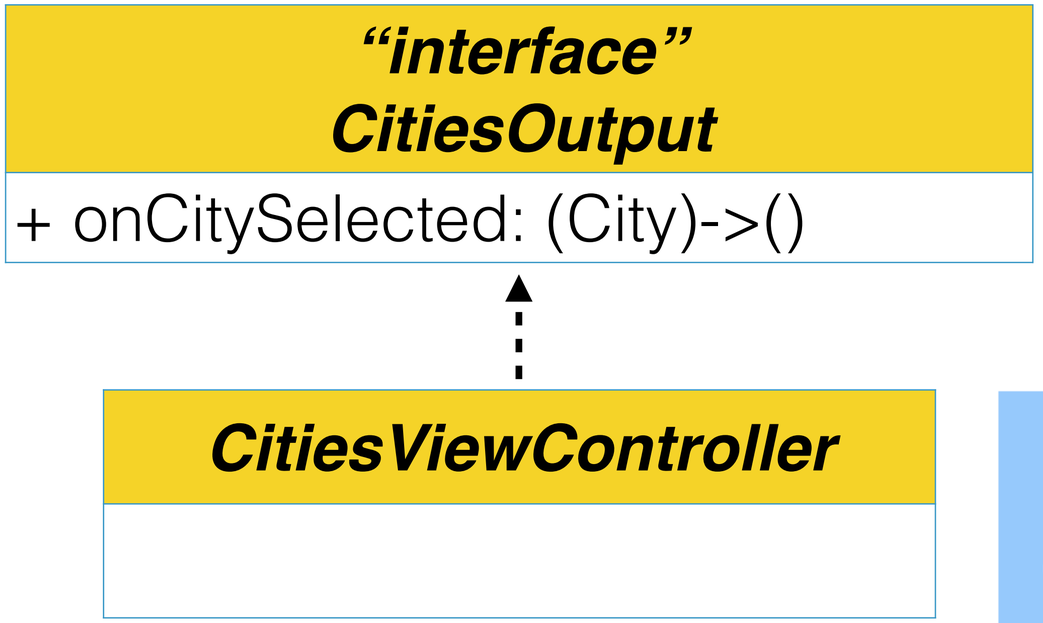


Apps Conf
App's 2017

```
protocol CitiesOutput {
    var onCitySelected: ((City) -> Void)? {
        get set
    }
}
```

| ***“interface” CitiesOutput*** |
| --- |
| + onCitySelected: (City)->() |

```
protocol CitiesOutput {
    var onCitySelected: ((City) -> Void)? {
        get set
    }
}


class CitiesViewController: CitiesOutput {
    var onCitySelected: ((City) -> Void)?
}
```




Avito

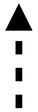


Apps Conf
App's 2017

| ***UserEditCoordinator*** |
| --- |
| + start() |
| + controllerFactory: ControllerFactory |

```
protocol CitiesOutput {
    var onCitySelected: ((City) -> Void)? {
        get set
    }
}


class CitiesViewController: CitiesOutput {
    var onCitySelected: ((City) -> Void)?
}
```

| ***“interface” CitiesOutput*** |
| --- |
| + onCitySelected: (City)->() |



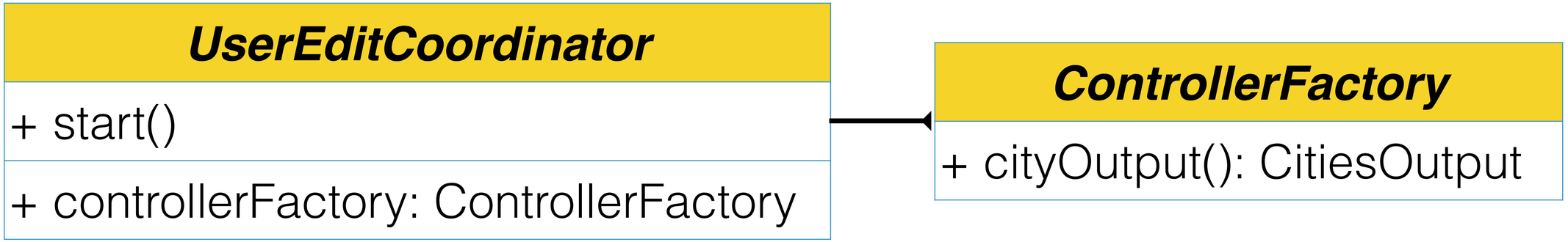
| ***CitiesViewController*** |
| --- |
| |

Avito

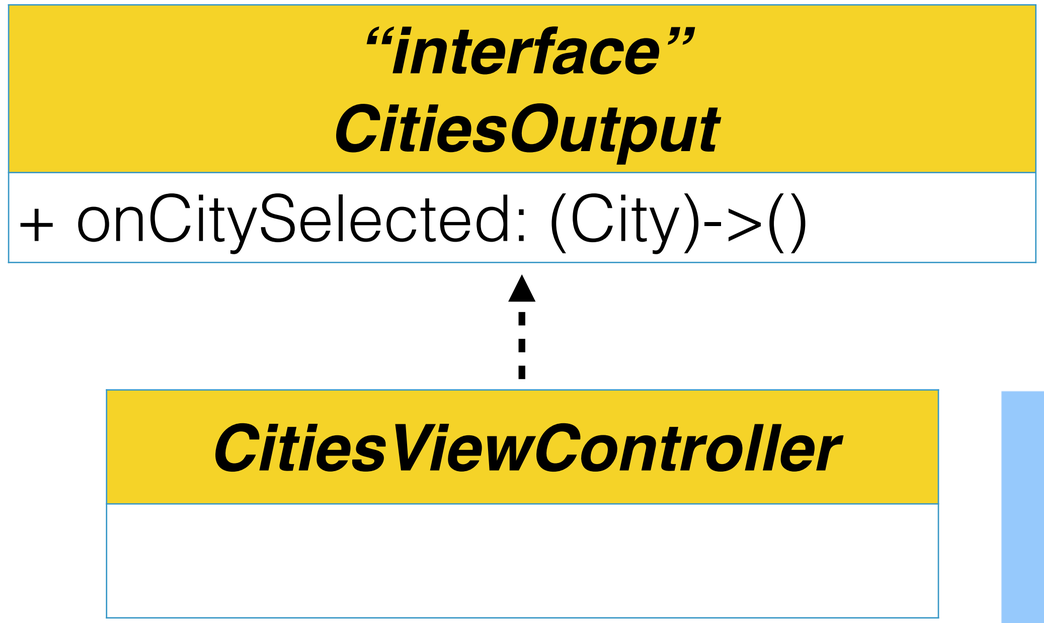


Apps Conf

App's 2017



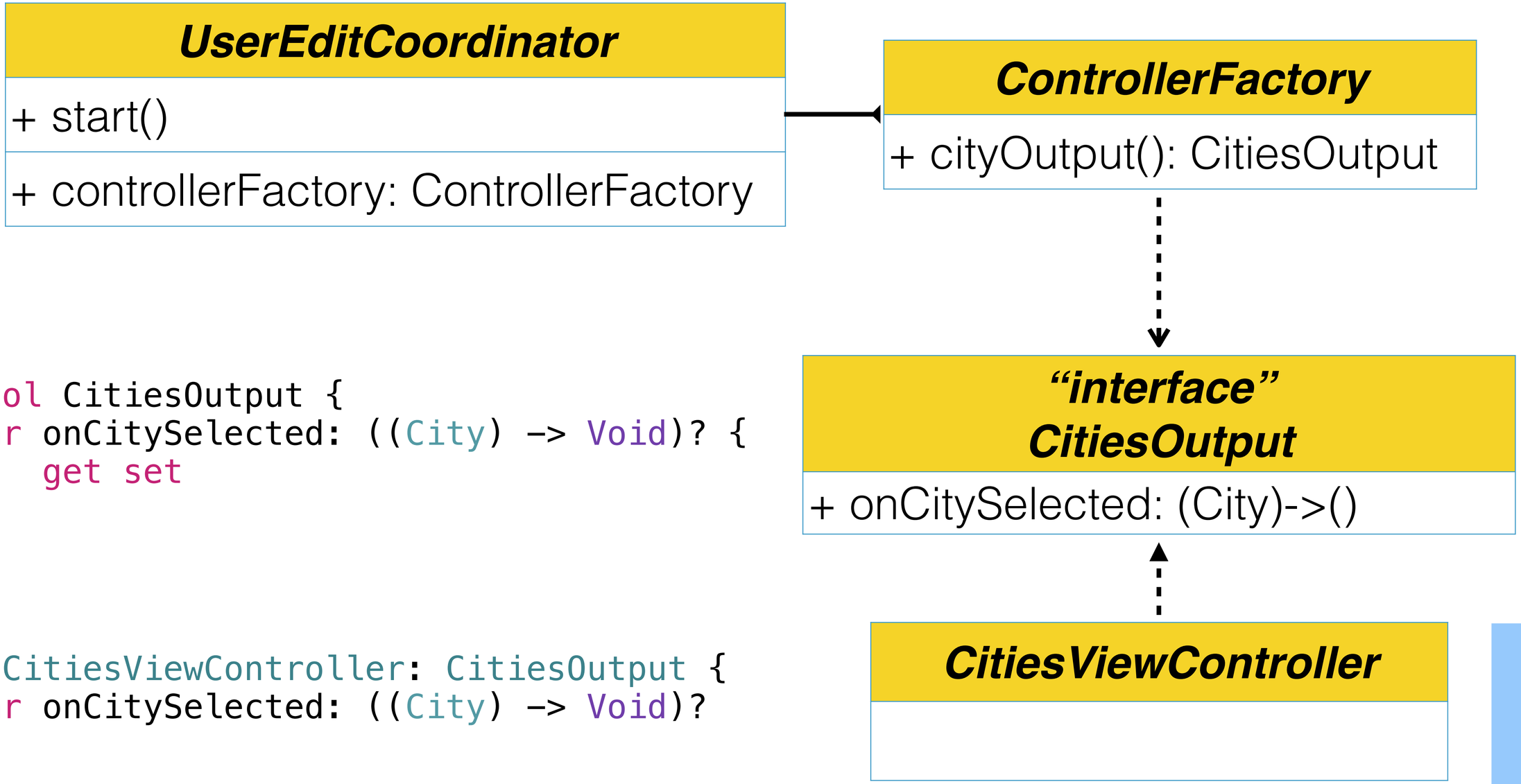

```
protocol CitiesOutput {
    var onCitySelected: ((City) -> Void)? {
        get set
    }
}


class CitiesViewController: CitiesOutput {
    var onCitySelected: ((City) -> Void)?
}
```




Avito



Apps Conf
App's 2017




```
protocol CitiesOutput {
    var onCitySelected: ((City) -> Void)? {
        get set
    }
}
```

```
class CitiesViewController: CitiesOutput {
    var onCitySelected: ((City) -> Void)?
}
```

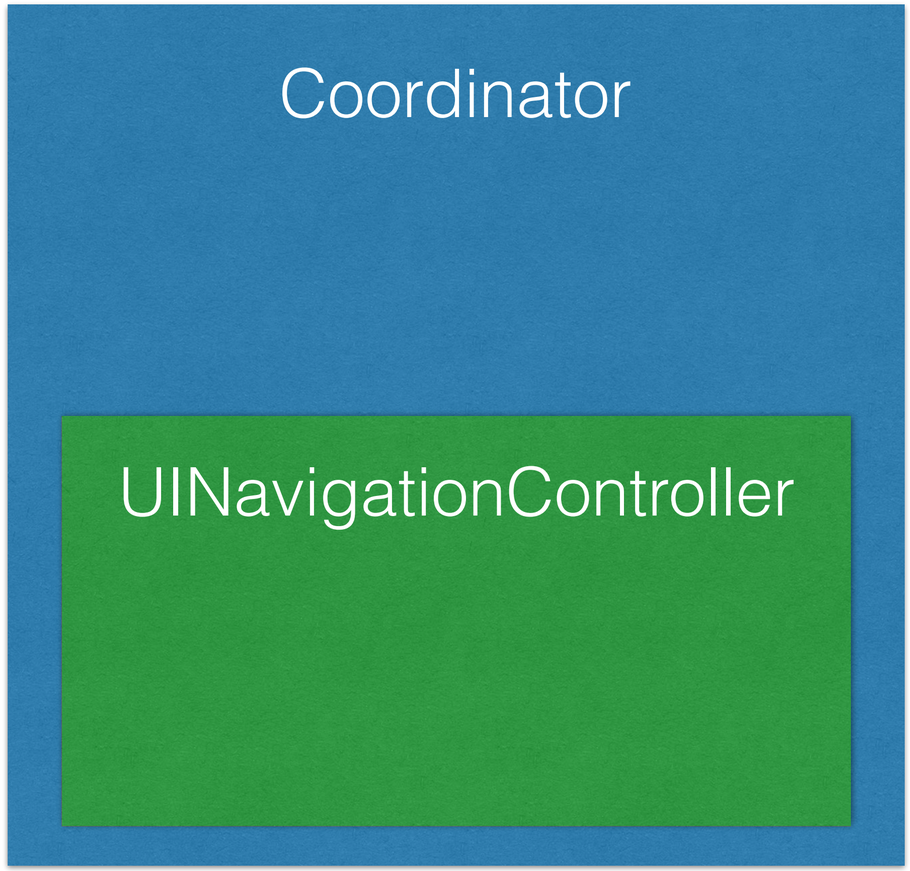
Avito



Apps Conf
App's 2017

# Composition VS Inheritance

# Composition VS Inheritance

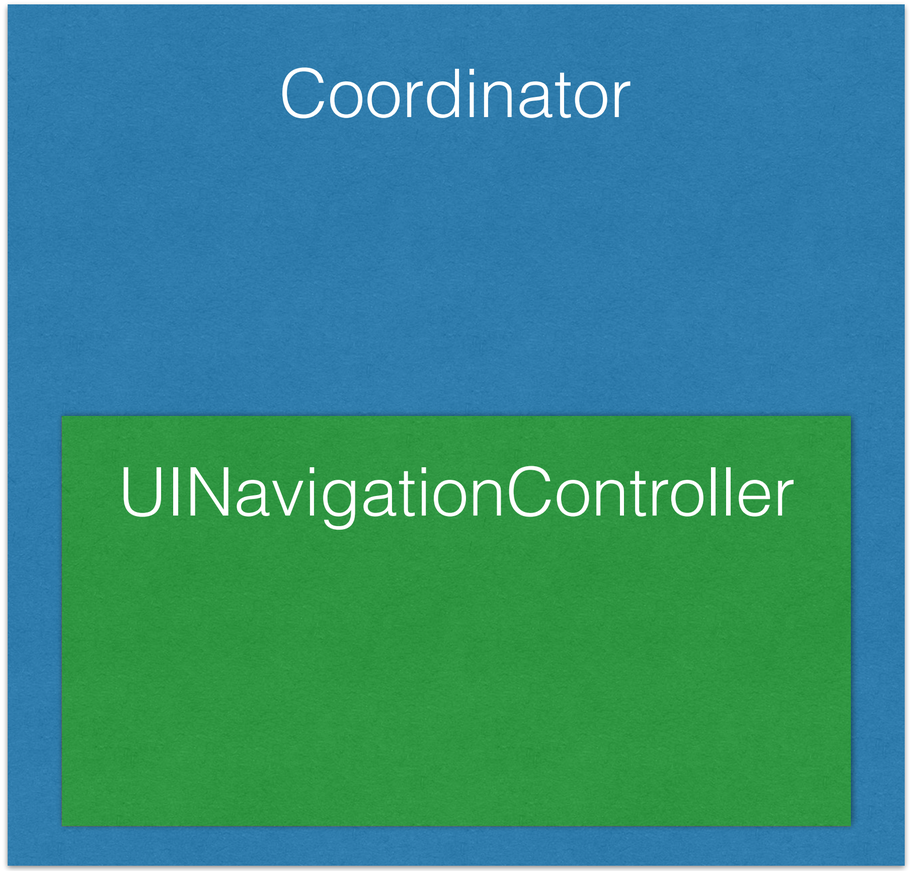



Avito

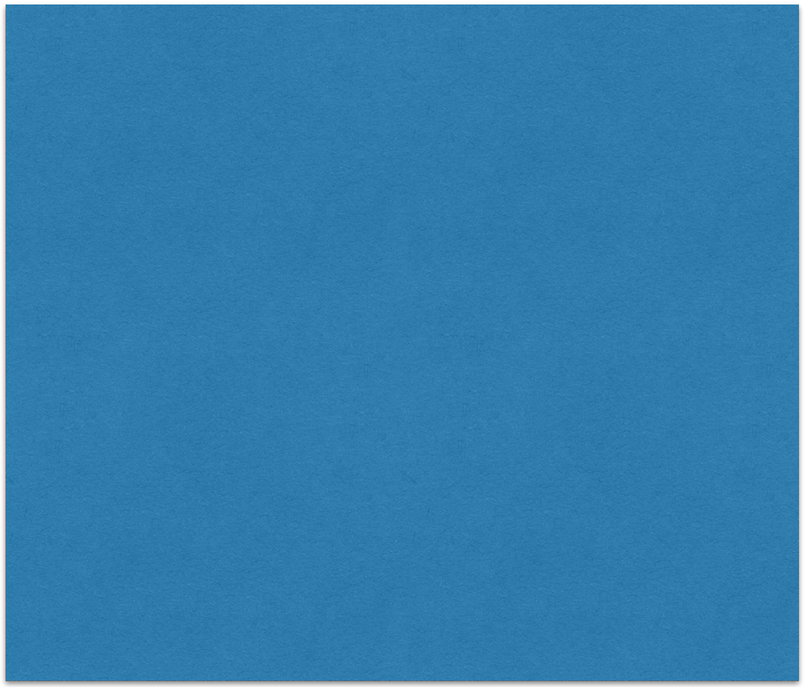


Apps Conf
App's 2017

# Composition VS Inheritance




Coordinator: UINavigationController



# Storyboards

# Storyboards

```
class BaseViewController: UIViewController {

    var onPrepareForSegue: ((UIStoryboardSegue, Any?) -> ())?

    override func prepare(for segue: UIStoryboardSegue,
                          sender: Any?) {

        onPrepareForSegue?(segue, sender)
    }
}
```

Avito



Apps Conf

# Storyboards

```
class BaseViewController: UIViewController {

    var onPrepareForSegue: ((UIStoryboardSegue, Any?) -> ())?

    override func prepare(for segue: UIStoryboardSegue,
                          sender: Any?) {

        onPrepareForSegue?(segue, sender)
    }
}
```



Avito

Apps Conf
App's 2017

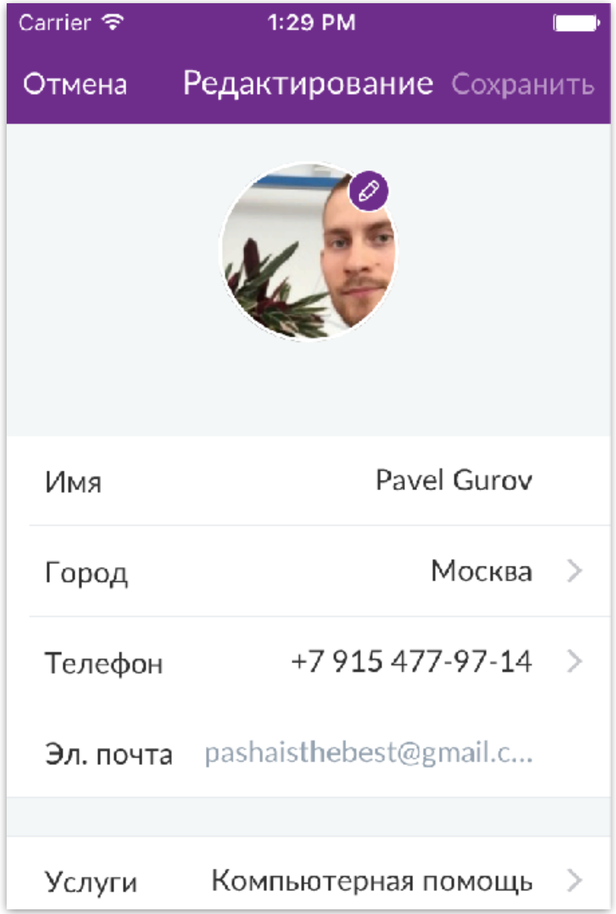
```
class StoryboardCoordinator<RootType: UIViewController> {

    let navigationController: UINavigationController
    let rootViewController: RootType

    init(storyboard: UIStoryboard) {
        navigationController = storyboard.initialNavigation
        rootViewController =
            navigationController!.topViewController as! RootType
    }
}

typealias UserEditCoordinator =
    StoryboardCoordinator<UserEditViewController>
```

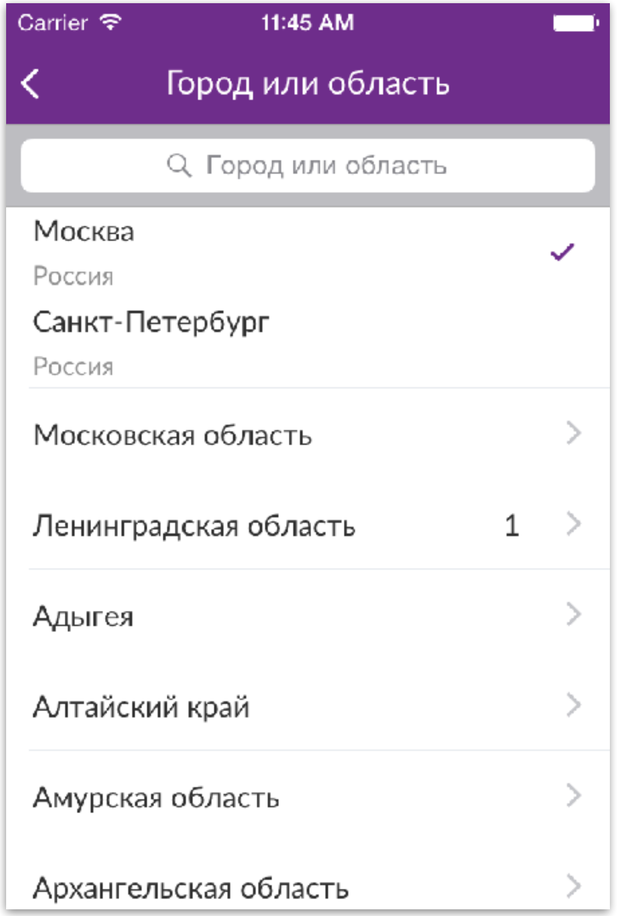
```
class StoryboardCoordinator<RootType: UIViewController> {

    let navigationController: UINavigationController
    let rootViewController: RootType

    init(storyboard: UIStoryboard) {
        navigationController = storyboard.initialNavigation
        rootViewController =
            navigationController!.topViewController as! RootType
    }
}

typealias UserEditCoordinator =
    StoryboardCoordinator<UserEditViewController>
```

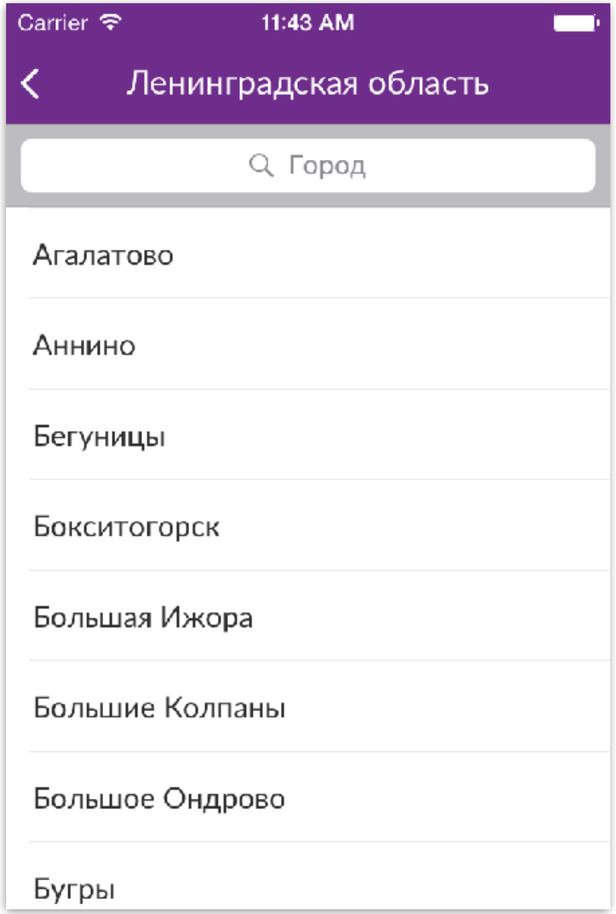
```
class StoryboardCoordinator<RootType: UIViewController> {

    let navigationController: UINavigationController
    let rootViewController: RootType

    init(storyboard: UIStoryboard) {
        navigationController = storyboard.initialNavigation
        rootViewController =
            navigationController!.topViewController as! RootType
    }
}

typealias UserEditCoordinator =
    StoryboardCoordinator<UserEditViewController>
```

```
class StoryboardCoordinator<RootType: UIViewController> {

    let navigationController: UINavigationController
    let rootViewController: RootType

    init(storyboard: UIStoryboard) {
        navigationController = storyboard.initialNavigation
        rootViewController =
            navigationController!.topViewController as! RootType
    }
}

typealias UserEditCoordinator =
    StoryboardCoordinator<UserEditViewController>
```

# Coordinator



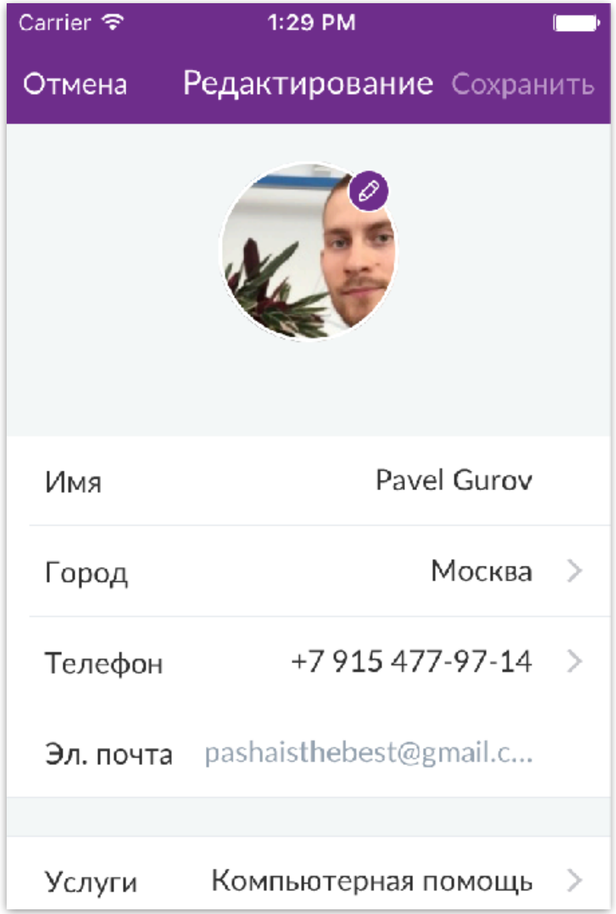



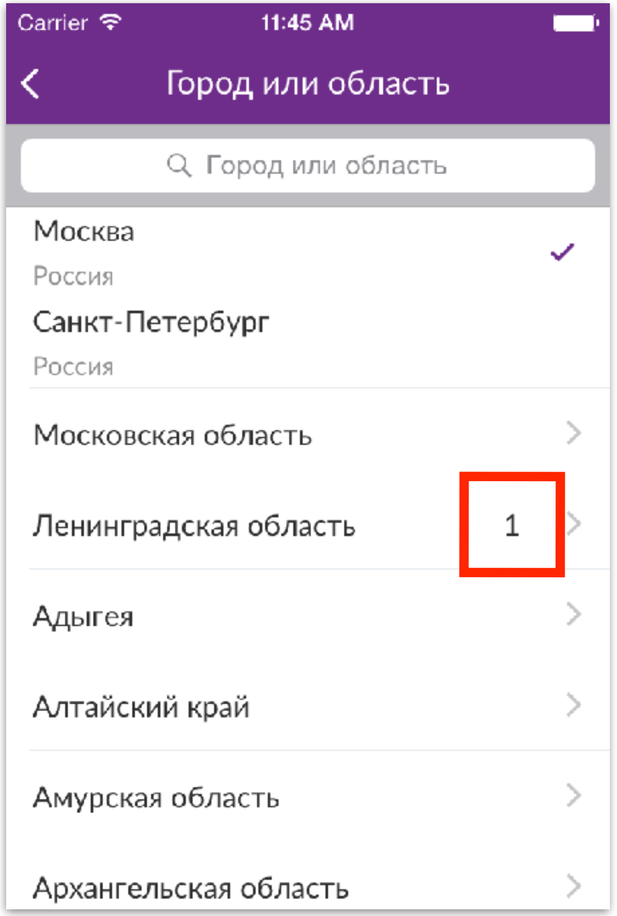

Avito

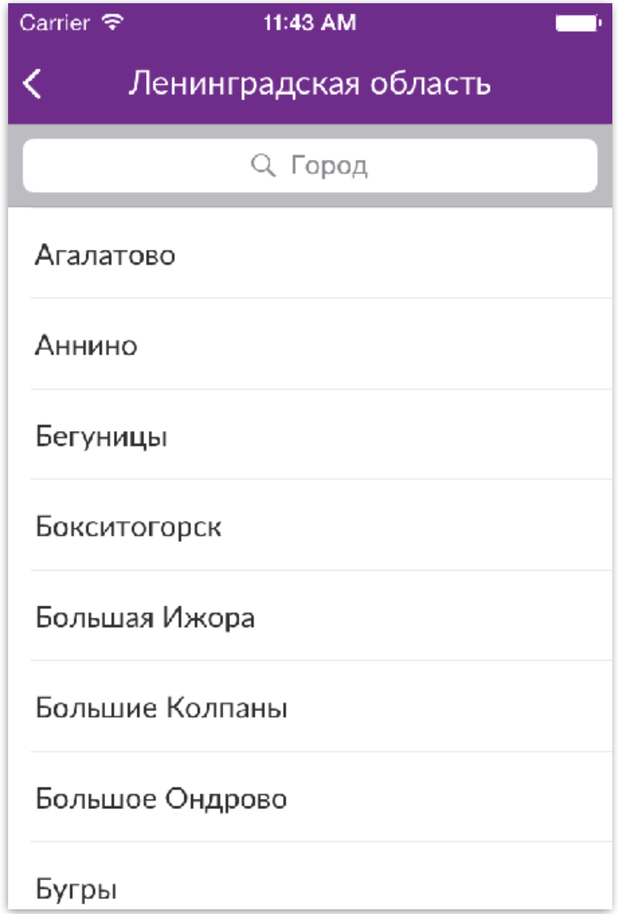


Apps Conf

# Coordinator



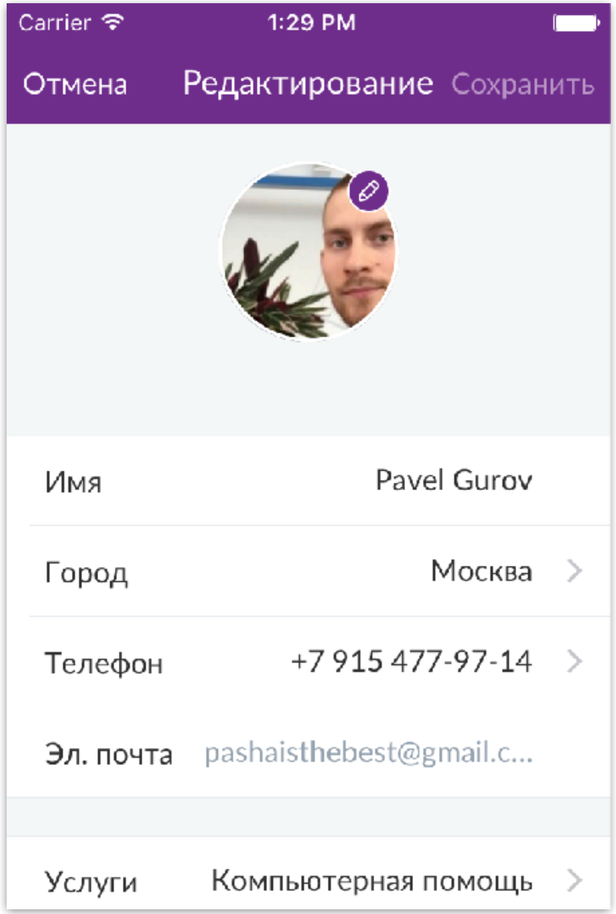



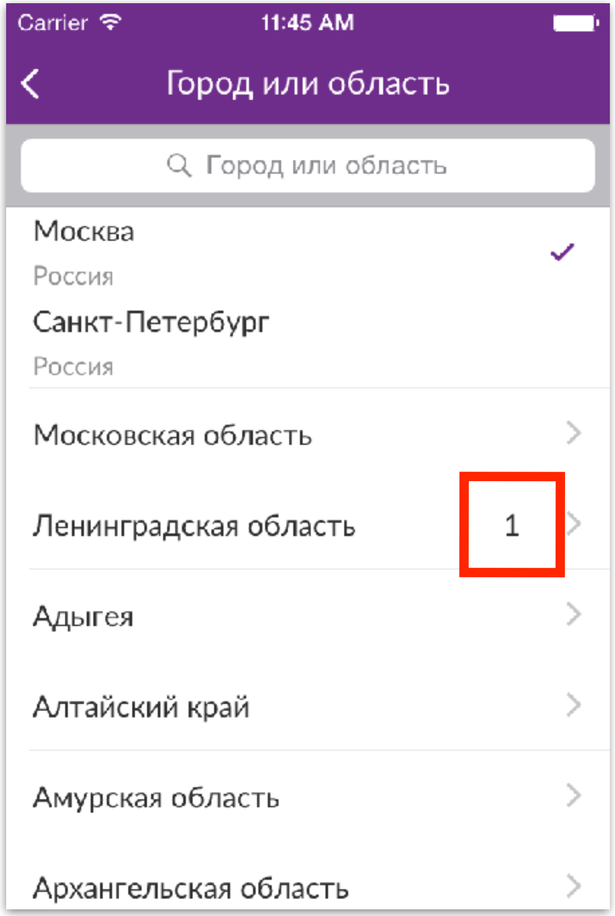

Avito

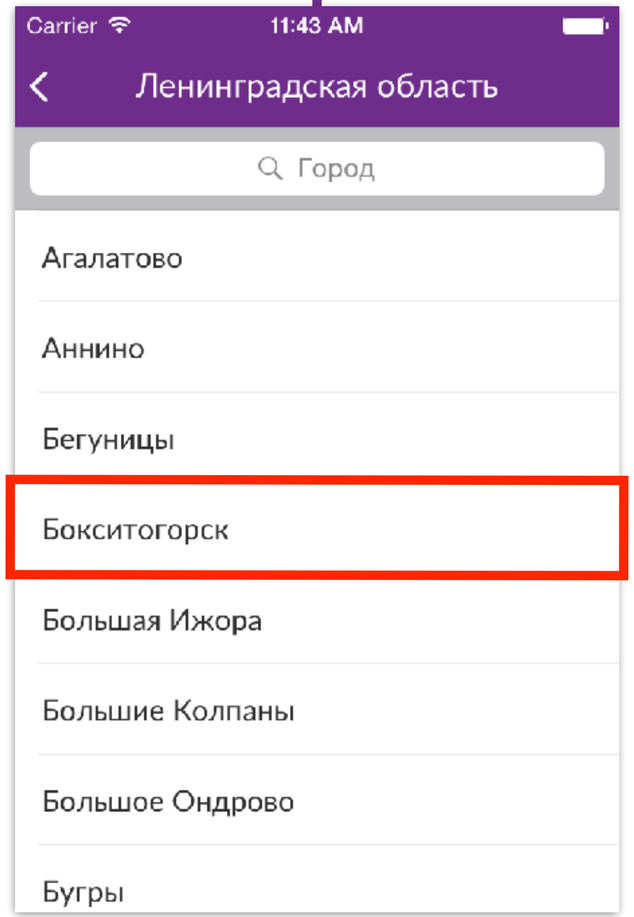


Apps Conf 2017

# Coordinator



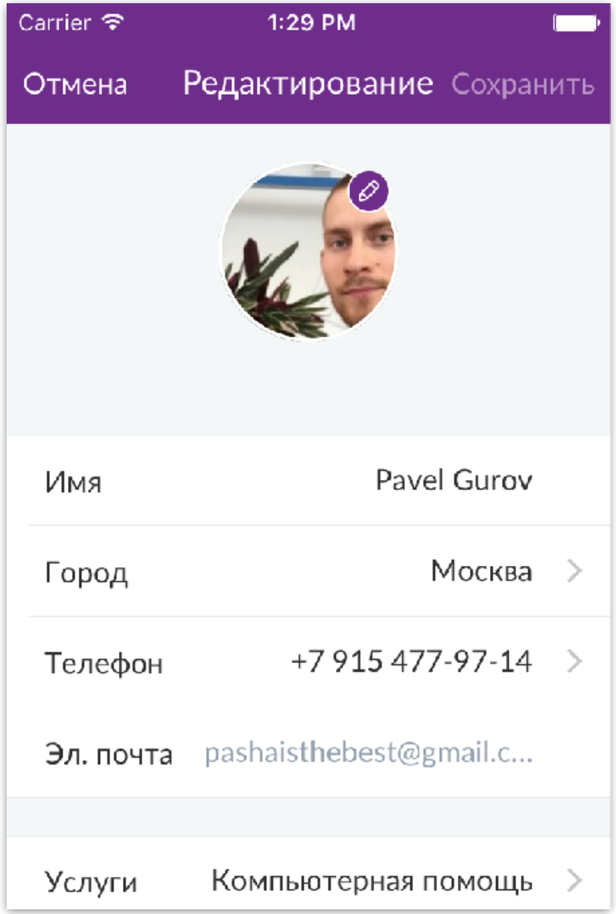



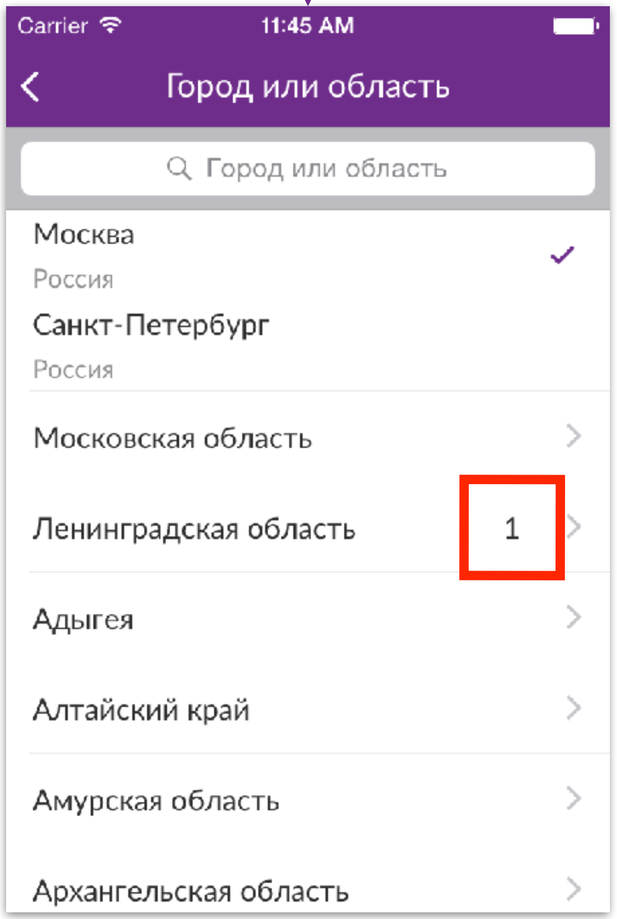

Avito

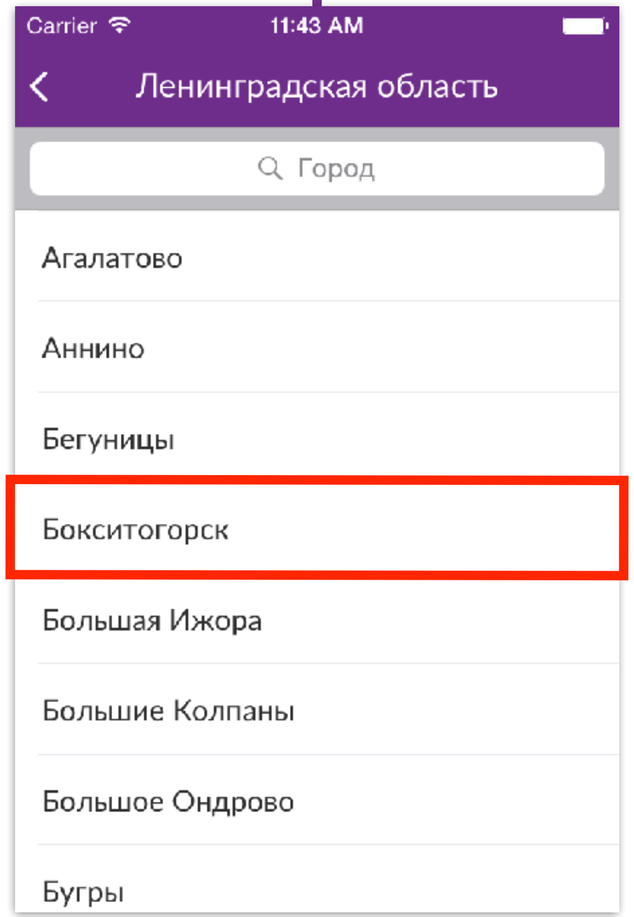


Apps Conf 2017

# Coordinator



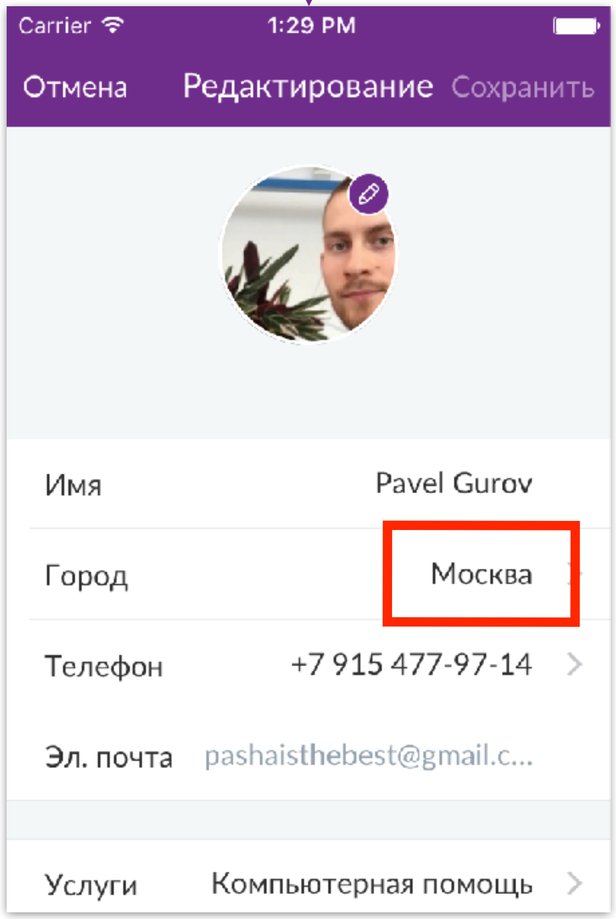



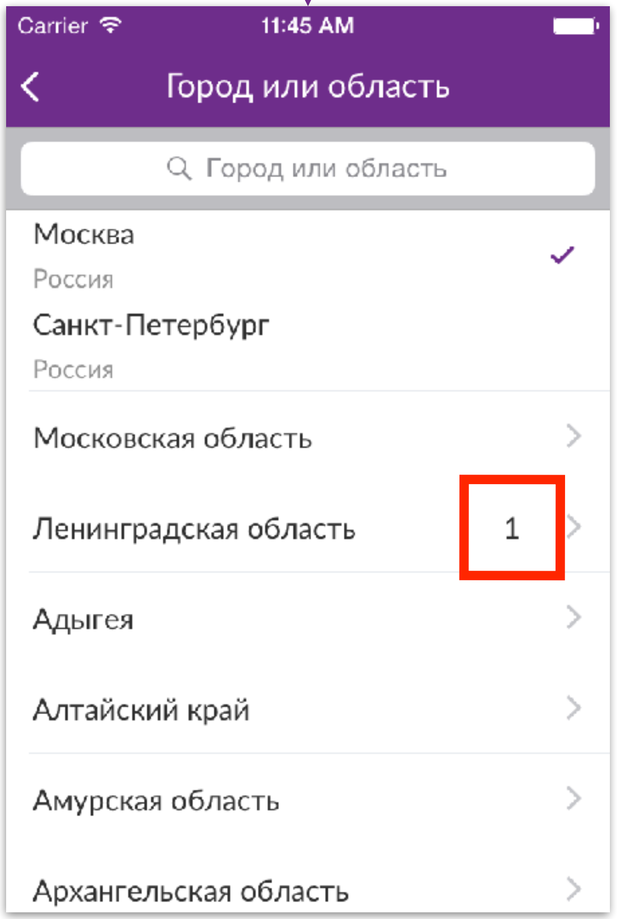

Avito

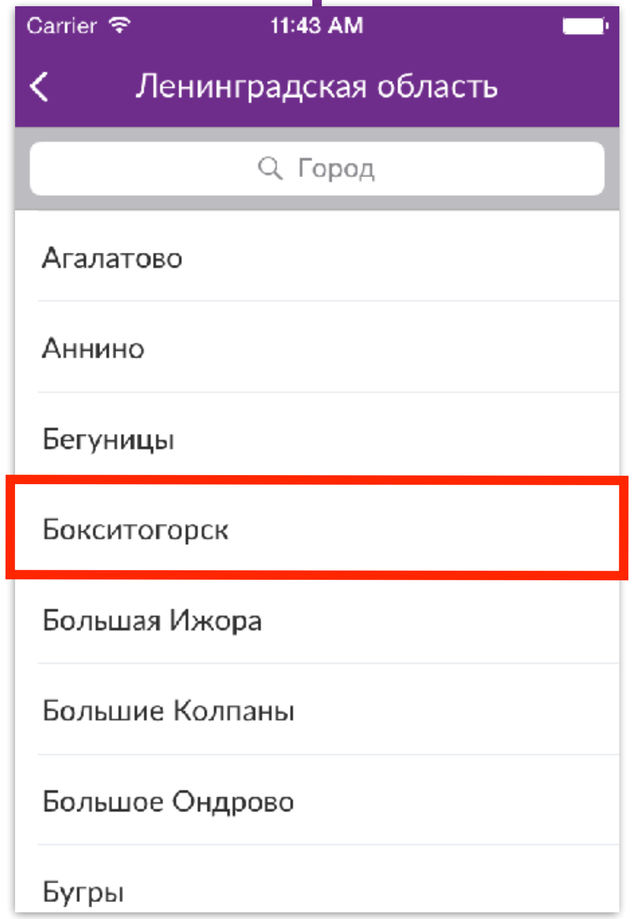


Apps Conf 2017

# Coordinator




Avito



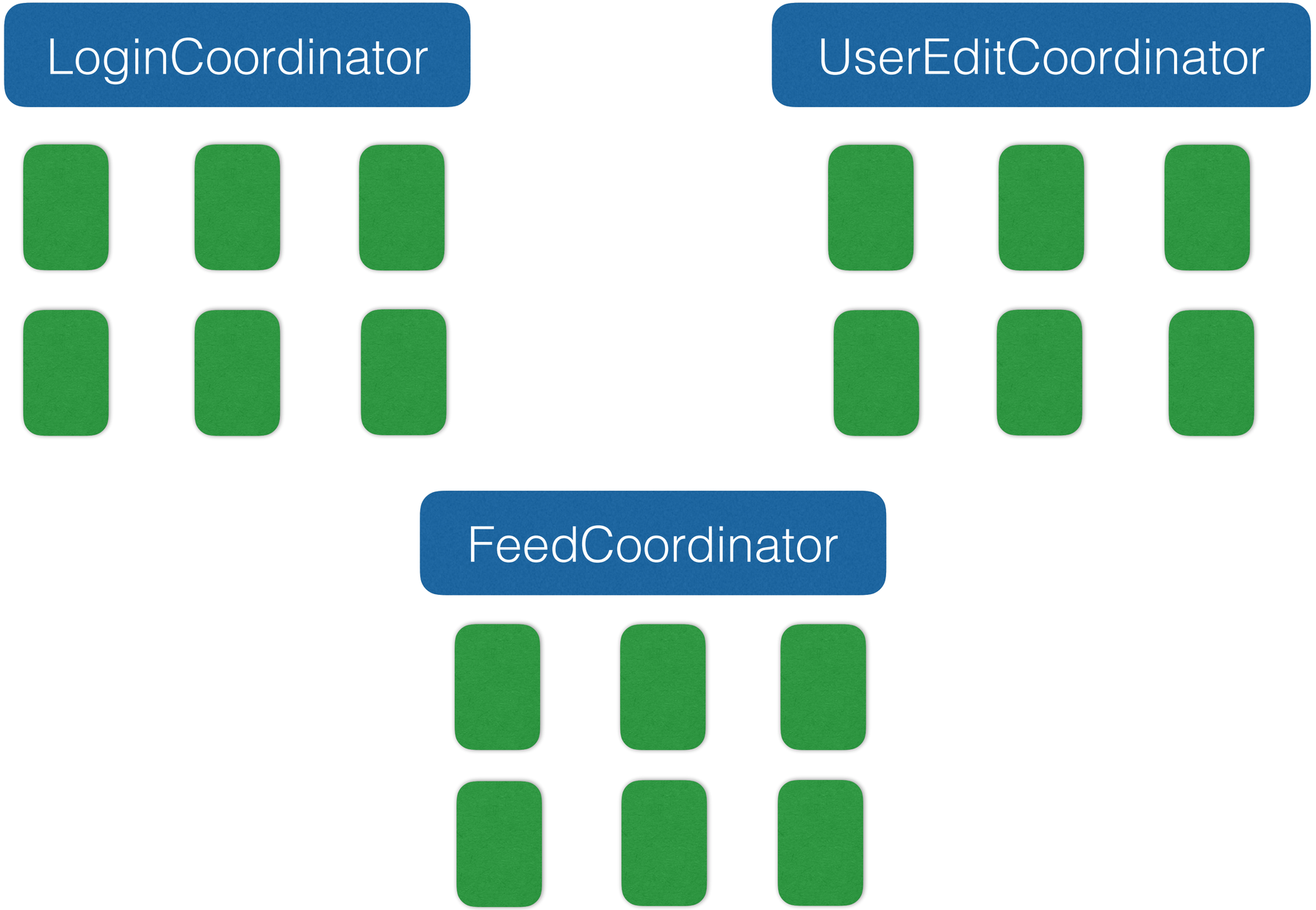
Apps Conf 2017

# Рефакторинг существующего кода

Avito



Apps Conf
App's 2017




Avito



Apps Conf
App's 2017

LoginCoordinator

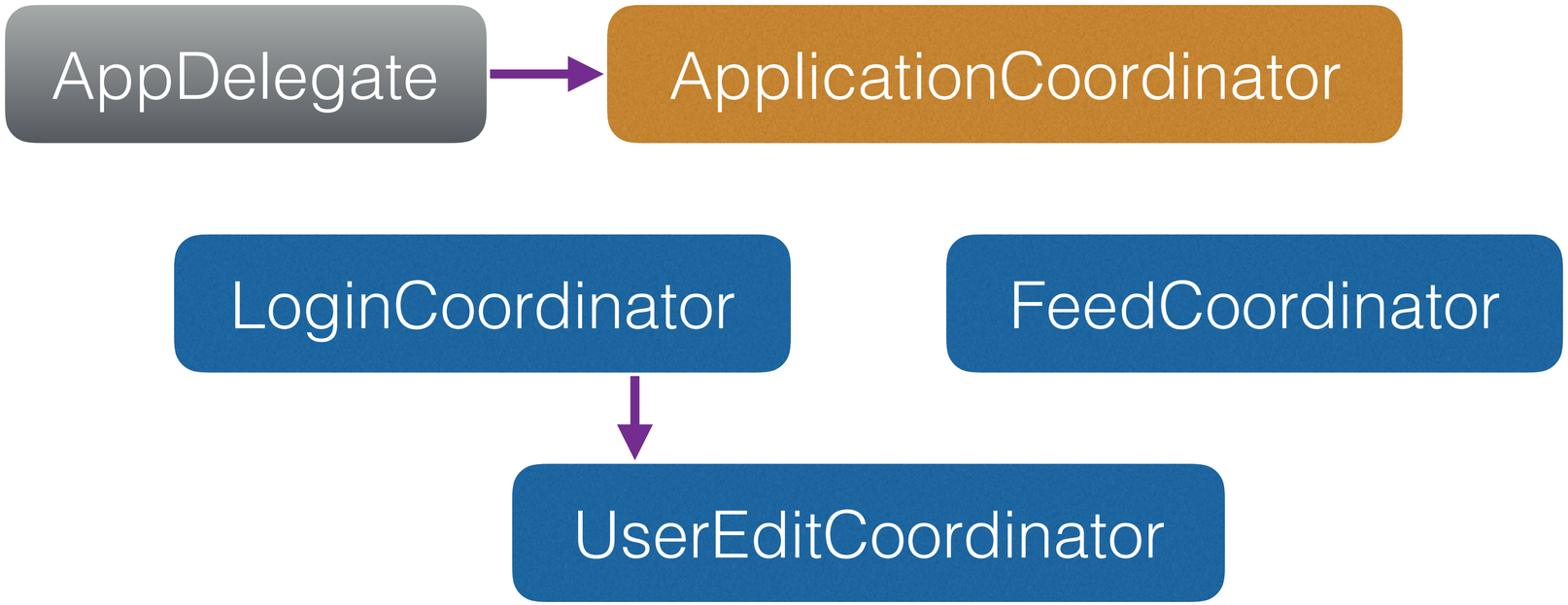
FeedCoordinator

UserEditCoordinator

Avito



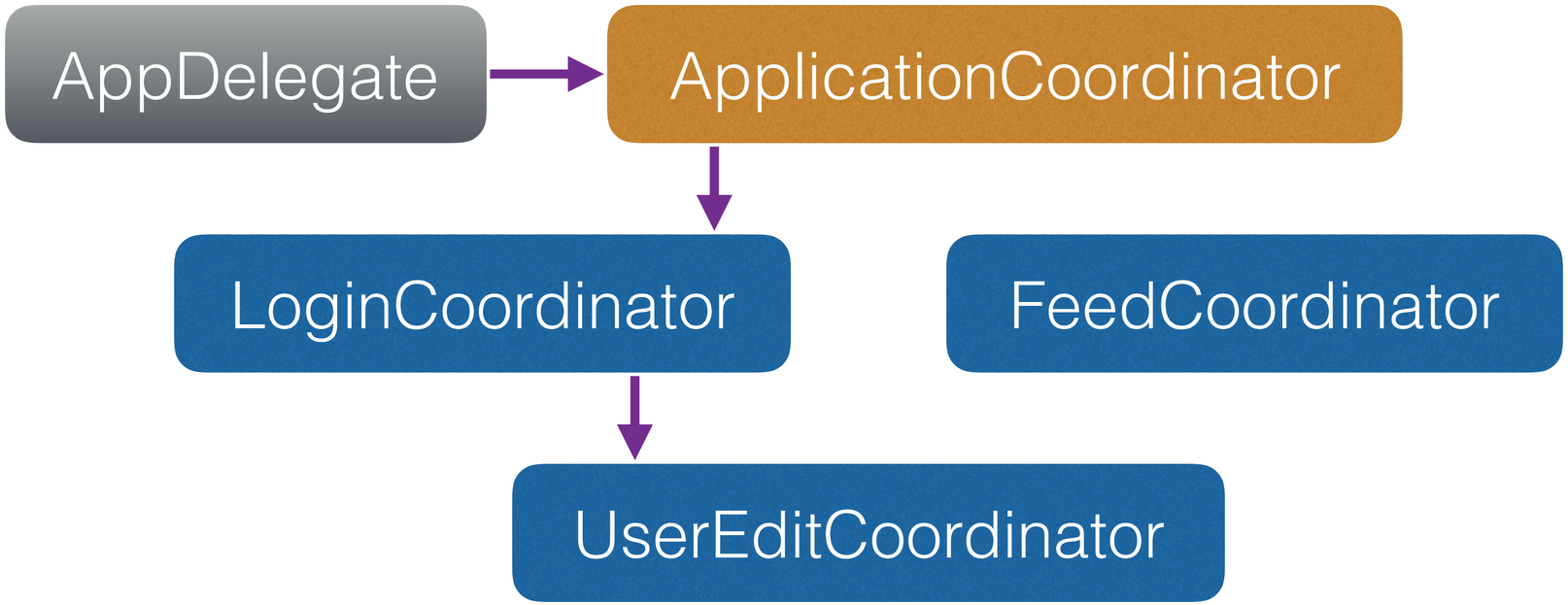
Apps Conf
App's 2017




Avito



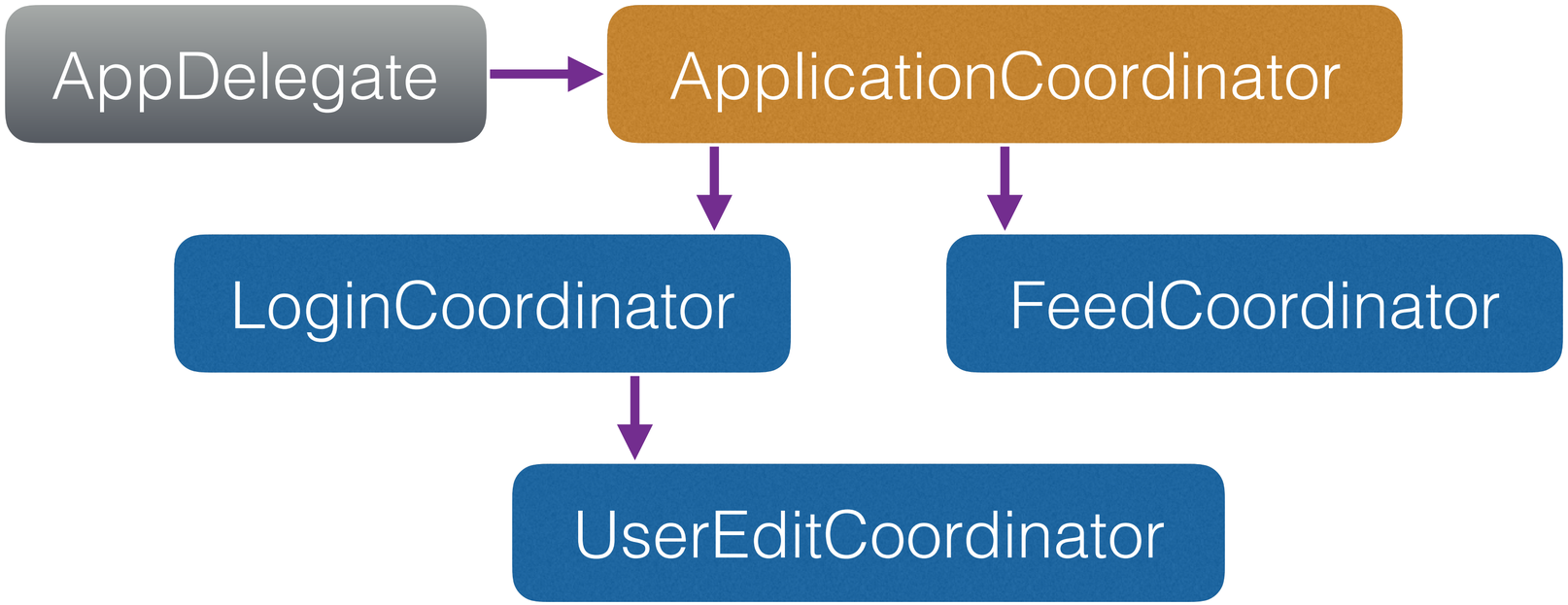
Apps Conf 2017




Avito



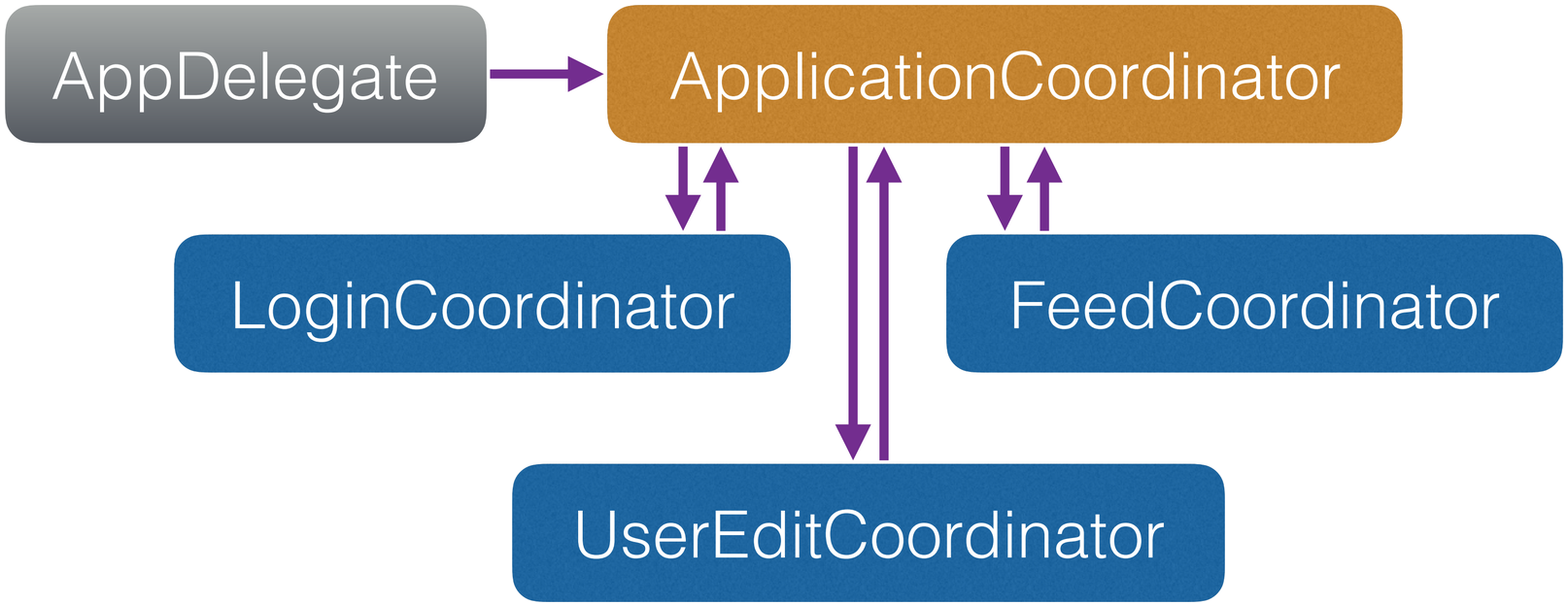
Apps Conf 2017




Avito



Apps Conf 2017




Avito



Apps Conf
App's 2017

# Временный старт Coordinator внутри UIViewController

```
class ViewController: UIViewController {

    var nextCoordinator: NextCoordinator?

    @IBAction func someAction(_ sender: UIButton) {
        nextCoordinator = NextCoordinator(
            navVC: navigationController
        )
        nextCoordinator?.start()
    }
}
```

Avito



Apps Conf 2017

# Итог

# Итог

- Независимые экраны и сценарии

Avito



Apps Conf

# Итог

- Независимые экраны и сценарии
- Легко менять порядок

# Итог

- Независимые экраны и сценарии
- Легко менять порядок
- Упрощается передача данных между экранами

# Итог

- Независимые экраны и сценарии
- Легко менять порядок
- Упрощается передача данных между экранами
- Никаких сторонних зависимостей =)

# Итог

- Независимые экраны и сценарии
- Легко менять порядок
- Упрощается передача данных между экранами
- Никаких сторонних зависимостей =)

github.com/pegurov/CoordinatorsDemo

pegurov@avito.ru

Avito



Apps Conf