GWT MVP Framework

- Для чего нужен MVP Framework
- Компоненты MVP
- Принципы MVP
- Пример: список контактов
- Загрузка данных в widget
- Вызов событий
- События и их обработка
- Регистрация обработчиков событий
- История переходов между экранными формами
- Компонента AppController

Для чего нужен MVP Framework?

1. Разделение проекта на независимые зоны ответственности в условиях одновременной работы нескольких разработчиков с исходным кодом проекта
2. Уменьшение покрытия кода автоматизированными тестами на основе GWTTestCase

Разделение реализуемой функциональности на логические компоненты.

Компоненты MVP

- Model – объекты бизнес логики, модели данных
- View – представление данных (элементы графического пользовательского интерфейса, экранные формы (карточки, таблицы))
- Presenter – логика приложения (последовательность переходов между экранными формами, взаимодействие клиента с сервером, обработка событий элементов графического интерфейса пользователя)
- AppController – логика приложения, не относящаяся к компоненте Presenter

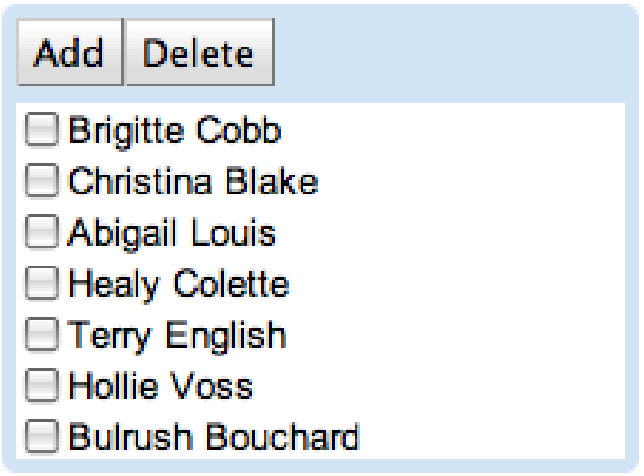
Принципы MVP

Слабое связывание логических компонент Model, View и Presenter

1. Изменение View не требует доработки Presenter
2. Изменение Model не требует доработки View

Пример: список контактов

Определение интерфейса Display




```
20  public class ContactsPresenter implements Presenter {
21
22    public interface Display {
23      HasClickHandlers getAddButton();
24      HasClickHandlers getDeleteButton();
25      HasClickHandlers getList();
26      void setData(List<String> data);
27      int getClickedRow(ClickEvent event);
28      List<Integer> getSelectedRows();
29      Widget asWidget();
30    }
31  }
```

```
34  private final ContactsServiceAsync rpcService;
35  private final HandlerManager eventBus;
36  private final Display display;
37
38  public ContactsPresenter(ContactsServiceAsync rpcService, HandlerManager eventBus, Display view) {
39    this.rpcService = rpcService;
40    this.eventBus = eventBus;
41    this.display = view;
42  }
```

Загрузка данных в widget

```
101  private void fetchContactDetails() {
102    rpcService.getContactDetails(new AsyncCallback<ArrayList<ContactDetails>>() {
103      public void onSuccess(ArrayList<ContactDetails> result) {
104          contactDetails = result;
105          sortContactDetails();
106          List<String> data = new ArrayList<String>();
107
108          for (int i = 0; i < result.size(); ++i) {
109            data.add(contactDetails.get(i).getDisplayName());
110          }
111
112          display.setData(data);
113      }
114
115      public void onFailure(Throwable caught) {
116        Window.alert("Error fetching contact details");
117      }
118    });
119  }
```

Вызов событий

```
44  public void bind() {
45    display.getAddButton().addClickHandler(new ClickHandler() {
46      public void onClick(ClickEvent event) {
47        eventBus.fireEvent(new AddContactEvent());
48      }
49    });
50
51    display.getDeleteButton().addClickHandler(new ClickHandler() {
52      public void onClick(ClickEvent event) {
53        deleteSelectedContacts();
54      }
55    });
56
57    display.getList().addClickHandler(new ClickHandler() {
58      public void onClick(ClickEvent event) {
59        int selectedRow = display.getClickedRow(event);
60
61        if (selectedRow >= 0) {
62          String id = contactDetails.get(selectedRow).getId();
63          eventBus.fireEvent(new EditContactEvent(id));
64        }
65      }
66    });
67  }
```

```
121   private void deleteSelectedContacts() {
122     List<Integer> selectedRows = display.getSelectedRows();
123     ArrayList<String> ids = new ArrayList<String>();
124
125     for (int i = 0; i < selectedRows.size(); ++i) {
126       ids.add(contactDetails.get(selectedRows.get(i)).getId());
127     }
128
129     rpcService.deleteContacts(ids, new AsyncCallback<ArrayList<ContactDetails>>() {
130       public void onSuccess(ArrayList<ContactDetails> result) {
131         contactDetails = result;
132         sortContactDetails();
133         List<String> data = new ArrayList<String>();
134
135         for (int i = 0; i < result.size(); ++i) {
136           data.add(contactDetails.get(i).getDisplayName());
137         }
138
139         display.setData(data);
140
141       }
142
143       public void onFailure(Throwable caught) {
144         Window.alert("Error deleting selected contacts");
145       }
146     });
147   }
148 }
```

События и их обработка

EditContactEvent

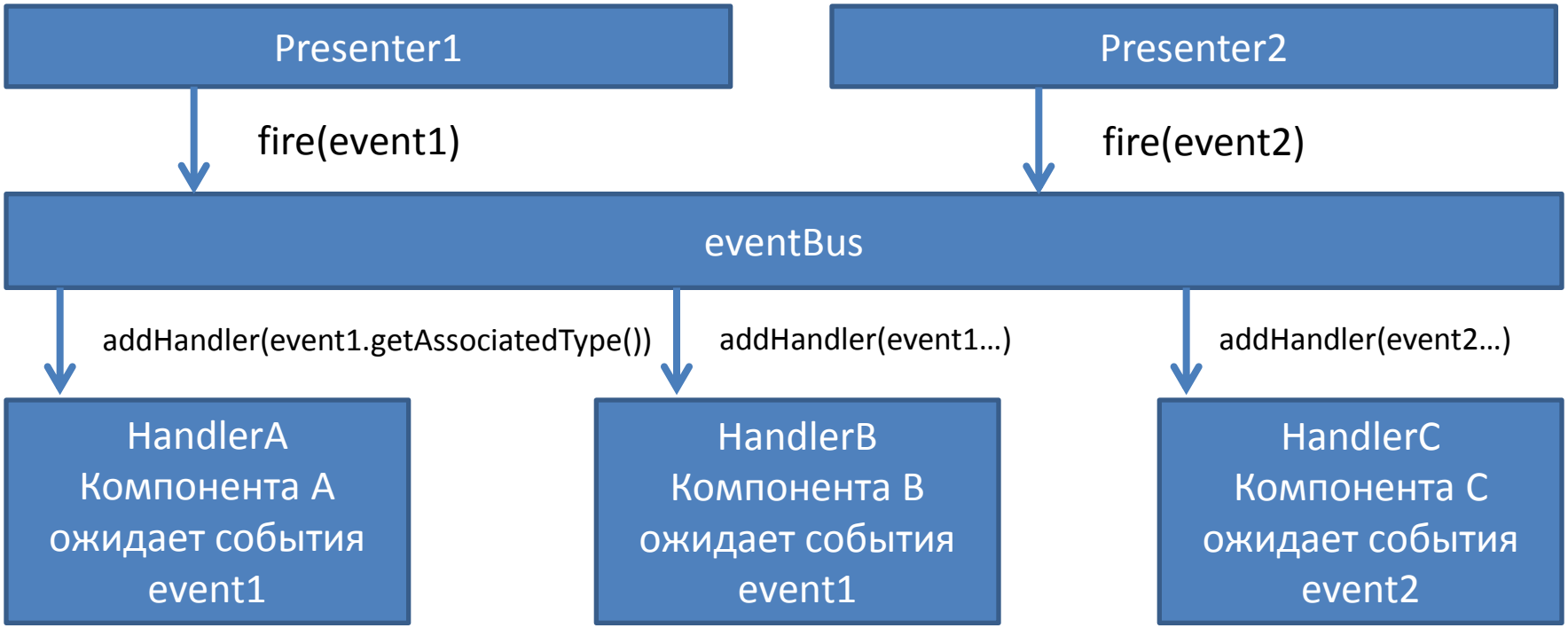
События регистрируемые Event Bus должны наследоваться от абстрактного класса com.google.gwt.event.shared.GwtEvent<H>, определяя методы dispatch() и getAssciatedType()

EditContactEventHandler

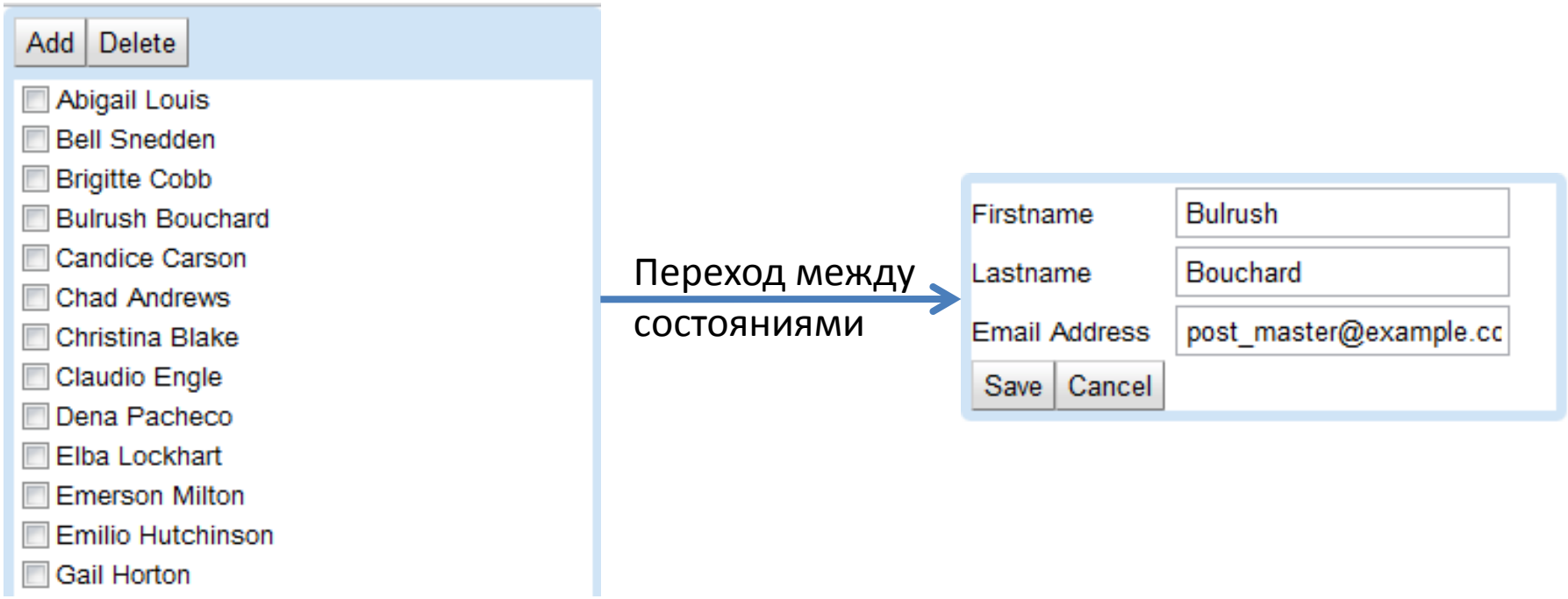
Обработчики событий должны наследоваться от com.google.gwt.event.shared.EventHandler

Регистрация обработчика события

```
44      eventBus.addHandler(EditContactEvent.TYPE,
45          new EditContactEventHandler() {
46            public void onEditContact(EditContactEvent event) {
47              doEditContact(event.getId());
48            }
49          });
```



История переходов между экранными формами



Состояние: “list”

Состояние: “edit”

Компонента AppController

AppController

AppController должен реализовывать интерфейс ValueChangeHandler, определяя метод onValueChange(), и быть зарегистрирован History.addValueChangeHandler()

```
95  public void onValueChange(ValueChangeEvent<String> event) {
96    String token = event.getValue();
97
98    if (token != null) {
99      Presenter presenter = null;
100
101     if (token.equals("list")) {
102       presenter = new ContactsPresenter(rpcService, eventBus, new ContactsView());
103     }
104     else if (token.equals("add")) {
105       presenter = new EditContactPresenter(rpcService, eventBus, new EditContactView());
106     }
107     else if (token.equals("edit")) {
108       presenter = new EditContactPresenter(rpcService, eventBus, new EditContactView());
109     }
110
111     if (presenter != null) {
112       presenter.go(container);
113     }
114   }
115 }
```

Узнать больше

1. http://www.google.com/intl/ru-RU/events/io/2009/sessions/GoogleWebToolkitBestPractices.html
2. http://code.google.com/intl/en/webtoolkit/doc/latest/DevGuideMvpActivitiesAndPlaces.html