INTRODUCTION TO VAADIN 8

HAIJIAN WANG

@haijian_wang

vaadin}>

vaadin}>

Vaadin

#1 JAVA WEB UI FRAMEWORK

1. 100% Java
2. Strives for developer productivity
3. Big set of Components + Add-ons
4. Open Source
5. And much more…

Session’s content

- The Good Ol’ Boy - Vaadin 7
- The Brand New Field Model of Vaadin 8
- CustomField<T>
- Where are my Items & Properties?
- Being Lazy & Happy without Containers

Session’s content

- **The Good Ol’ Boy - Vaadin 7**
- The Brand New Field Model of Vaadin 8
- CustomField<T>
- Where are my Items & Properties?
- Being Lazy & Happy without Containers

How would you edit a customer record?

VAADIN 7

VAADIN 7

No data binding

```
@Override
 protected void init(VaadinRequest vaadinRequest) {
    VerticalLayout layout = new VerticalLayout();

    TextField firstName = new TextField("Firstname");
    TextField lastName = new TextField("Lastname");
```

VAADIN 7

No data binding

```
@Override
 protected void init(VaadinRequest vaadinRequest) {
    VerticalLayout layout = new VerticalLayout();

    TextField firstName = new TextField("Firstname");
    TextField lastName = new TextField("Lastname");

    CustomerDTO customer = new CustomerDTO();
```

VAADIN 7

No data binding

```
@Override
 protected void init(VaadinRequest vaadinRequest) {
    VerticalLayout layout = new VerticalLayout();

    TextField firstName = new TextField("Firstname");
    TextField lastName = new TextField("Lastname");

    CustomerDTO customer = new CustomerDTO();

    firstName.addValueChangeListener(new ValueChangeListener() {
       @Override
       public void valueChange(ValueChangeEvent event) {
          customer.setFirstName(firstName.getValue());
       }
    });
```

VAADIN 7

No data binding

```
@Override
 protected void init(VaadinRequest vaadinRequest) {
    VerticalLayout layout = new VerticalLayout();

    TextField firstName = new TextField("Firstname");
    TextField lastName = new TextField("Lastname");

    CustomerDTO customer = new CustomerDTO();

    firstName.addValueChangeListener(new ValueChangeListener() {
       @Override
       public void valueChange(ValueChangeEvent event) {
          customer.setFirstName(firstName.getValue());
       }
    });

    lastName.addValueChangeListener(new ValueChangeListener() {
       @Override
       public void valueChange(ValueChangeEvent event) {
          customer.setLastName(lastName.getValue());
       }
    });
```

VAADIN 7

No data binding

```
@Override
 protected void init(VaadinRequest vaadinRequest) {
    VerticalLayout layout = new VerticalLayout();

    TextField firstName = new TextField("Firstname");
    TextField lastName = new TextField("Lastname");

    CustomerDTO customer = new CustomerDTO();

    firstName.addValueChangeListener(new ValueChangeListener() {
       @Override
       public void valueChange(ValueChangeEvent event) {
          customer.setFirstName(firstName.getValue());
       }
    });

    lastName.addValueChangeListener(new ValueChangeListener() {
       @Override
       public void valueChange(ValueChangeEvent event) {
          customer.setLastName(lastName.getValue());
       }
    });

    layout.addComponents(firstName, lastName);

    setContent(layout);
 }
```

WHAT’S WRONG?

- Setters used explicitly
- Every Field has a Listener
- No way to Save / Cancel
- Lots of code
- Prone to error and change
- Invalid values not prevented

```
@Override
 protected void init(VaadinRequest vaadinRequest) {
    VerticalLayout layout = new VerticalLayout();

    TextField firstName = new TextField("Firstname");
    TextField lastName = new TextField("Lastname");

    CustomerDTO customer = new CustomerDTO();

    firstName.addValueChangeListener(new ValueChangeListener() {
       @Override
       public void valueChange(ValueChangeEvent event) {
          customer.setFirstName(firstName.getValue());
       }
    });

    lastName.addValueChangeListener(new ValueChangeListener() {
       @Override
       public void valueChange(ValueChangeEvent event) {
          customer.setLastName(lastName.getValue());
       }
    });

    layout.addComponents(firstName, lastName);

    setContent(layout);
 }
```

LET’S IMPROVE!

```
@Override
protected void init(VaadinRequest vaadinRequest) {
   VerticalLayout layout = new VerticalLayout();

   TextField firstName = new TextField("Firstname");
   TextField lastName = new TextField("Lastname");

   CustomerDTO customer = new CustomerDTO();
```

LET’S IMPROVE!

```
@Override
protected void init(VaadinRequest vaadinRequest) {
   VerticalLayout layout = new VerticalLayout();

   TextField firstName = new TextField("Firstname");
   TextField lastName = new TextField("Lastname");

   CustomerDTO customer = new CustomerDTO();

   Button save = new Button("Save", new Button.ClickListener() {

      @Override
      public void buttonClick(ClickEvent event) {
         customer.setFirstName(firstName.getValue());
         customer.setLastName(lastName.getValue());
      }
   });

   layout.addComponents(firstName, lastName, save);

   setContent(layout);
}
```

BENEFITS?

- Centralized value setting
- Validation could be added

```
@Override
protected void init(VaadinRequest vaadinRequest) {
   VerticalLayout layout = new VerticalLayout();

   TextField firstName = new TextField("Firstname");
   TextField lastName = new TextField("Lastname");

   CustomerDTO customer = new CustomerDTO();

   Button save = new Button("Save", new Button.ClickListener() {

      @Override
      public void buttonClick(ClickEvent event) {
         customer.setFirstName(firstName.getValue());
         customer.setLastName(lastName.getValue());
      }
   });

   layout.addComponents(firstName, lastName, save);

   setContent(layout);
}
```

WHAT’S WRONG?

- Setters used explicitly
- Lots of code
- Prone to error and change
- Hard to Validate

```
@Override
protected void init(VaadinRequest vaadinRequest) {
   VerticalLayout layout = new VerticalLayout();

   TextField firstName = new TextField("Firstname");
   TextField lastName = new TextField("Lastname");

   CustomerDTO customer = new CustomerDTO();

   Button save = new Button("Save", new Button.ClickListener() {

      @Override
      public void buttonClick(ClickEvent event) {
         customer.setFirstName(firstName.getValue());
         customer.setLastName(lastName.getValue());
      }
   });

   layout.addComponents(firstName, lastName, save);

   setContent(layout);
}
```

VAADIN 7

With FieldGroup

```
@Override
protected void init(VaadinRequest vaadinRequest) {
    VerticalLayout layout = new VerticalLayout();

    TextField firstName = new TextField("Firstname");
    TextField lastName = new TextField("Lastname");

    CustomerDTO customer = new CustomerDTO();
```

VAADIN 7

With FieldGroup

```
@Override
protected void init(VaadinRequest vaadinRequest) {
    VerticalLayout layout = new VerticalLayout();

    TextField firstName = new TextField("Firstname");
    TextField lastName = new TextField("Lastname");

    CustomerDTO customer = new CustomerDTO();

    FieldGroup fieldGroup = new FieldGroup();
    fieldGroup.bind(firstName, "firstName");
    fieldGroup.bind(lastName, "lastName");
    fieldGroup.setItemDataSource(new
                    BeanItem<CustomerDTO>(customer));
```

VAADIN 7

With FieldGroup

```
@Override
protected void init(VaadinRequest vaadinRequest) {
  VerticalLayout layout = new VerticalLayout();

  TextField firstName = new TextField("Firstname");
  TextField lastName = new TextField("Lastname");

  CustomerDTO customer = new CustomerDTO();

  FieldGroup fieldGroup = new FieldGroup();
  fieldGroup.bind(firstName, "firstName");
  fieldGroup.bind(lastName, "lastName");
  fieldGroup.setItemDataSource(new
              BeanItem<CustomerDTO>(customer));

  Button save = new Button("Save", new Button.ClickListener() {

    @Override
    public void buttonClick(ClickEvent event) {
      try {
        fieldGroup.commit();
      }
      catch(CommitException e) {
        // show errors
      }
    }
  });
```

BENEFITS?

- No more explicit setters
- Committable / Discardable
- Validation part of Commit

```java
@Override
protected void init(VaadinRequest vaadinRequest) {
  VerticalLayout layout = new VerticalLayout();

  TextField firstName = new TextField("Firstname");
  TextField lastName = new TextField("Lastname");

  CustomerDTO customer = new CustomerDTO();

  FieldGroup fieldGroup = new FieldGroup();
  fieldGroup.bind(firstName, "firstName");
  fieldGroup.bind(lastName, "lastName");
  fieldGroup.setItemDataSource(new
            BeanItem<CustomerDTO>(customer));

  Button save = new Button("Save", new Button.ClickListener() {

    @Override
    public void buttonClick(ClickEvent event) {
      try {
        fieldGroup.commit();
      }
      catch(CommitException e) {
        // show errors
      }
    }
  });
```

WHAT'S WRONG?

- Not type safe
- Prone to error and change
- BeanItem boiler plate

```
@Override
protected void init(VaadinRequest vaadinRequest) {
  VerticalLayout layout = new VerticalLayout();

  TextField firstName = new TextField("Firstname");
  TextField lastName = new TextField("Lastname");

  CustomerDTO customer = new CustomerDTO();

  FieldGroup fieldGroup = new FieldGroup();
  fieldGroup.bind(firstName, "firstName");
  fieldGroup.bind(lastName, "lastName");
  fieldGroup.setItemDataSource(new
            BeanItem<CustomerDTO>(customer));

  Button save = new Button("Save", new Button.ClickListener() {

    @Override
    public void buttonClick(ClickEvent event) {
      try {
        fieldGroup.commit();
      }
      catch(CommitException e) {
        // show errors
      }
    }
  });
```

#FFS

FIGHT FOR SIMPLICITY ;)

Session’s content

- The Good Ol’ Boy - Vaadin 7
- **The Brand New Field Model of Vaadin 8**
- CustomField<T>
- Where are my Items & Properties?
- Being Lazy & Happy without Containers

c.v.ui.AbstractField<T>

VAADIN 7	VAADIN 8
1853 loc	213 loc

vaadin}>

FieldGroup

TO

Binder

How would you edit a customer record?

VAADIN 8

VAADIN 8

With Binder

```
TextField firstName = ...
TextField lastName = ...

CustomerDTO customer = ...

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);
```

VAADIN 8

With Binder

```
TextField firstName = …
TextField lastName = …

CustomerDTO customer = …

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.bind(firstName, CustomerDTO::getFirstname, CustomerDTO::setFirstname);
```

VAADIN 8

With Binder

```
TextField firstName = …
TextField lastName = …

CustomerDTO customer = …

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.bind(firstName, CustomerDTO::getFirstname, CustomerDTO::setFirstname);
binder.bind(lastName, CustomerDTO::getLastname, CustomerDTO::setLastname);
```

VAADIN 8

With Binder

```
TextField firstName = …
TextField lastName = …

CustomerDTO customer = …

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.bind(firstName, CustomerDTO::getFirstname, CustomerDTO::setFirstname);
binder.bind(lastName, CustomerDTO::getLastname, CustomerDTO::setLastname);

binder.readBean(customer);
```

VAADIN 8

Commit / Discard?

```
TextField firstName = …
TextField lastName = …

CustomerDTO customer = …

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.bind(firstName, CustomerDTO::getFirstname, CustomerDTO::setFirstname);
binder.bind(lastName, CustomerDTO::getLastname, CustomerDTO::setLastname);

binder.readBean(customer);
```

VAADIN 8

Commit / Discard?

```
TextField firstName = …
TextField lastName = …

CustomerDTO customer = …

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.bind(firstName, CustomerDTO::getFirstname, CustomerDTO::setFirstname);
binder.bind(lastName, CustomerDTO::getLastname, CustomerDTO::setLastname);

binder.readBean(customer);

Button save = new Button("Save" , e -> binder.writeBeanIfValid(customer));
Button discard = new Button("Discard" , e -> binder.readBean(customer));
```

Conversion

VAADIN 8

Conversion with Binding

```
TextField yearOfBirth = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);
```

VAADIN 8

Conversion with Binding

```
TextField yearOfBirth = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(yearOfBirth)
```

VAADIN 8

Conversion with Binding

```
TextField yearOfBirth = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(yearOfBirth)
    .withConverter(Integer::valueOf, String::valueOf)
```

VAADIN 8

Conversion with Binding

```
TextField yearOfBirth = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(yearOfBirth)
    .withConverter(Integer::valueOf, String::valueOf)
    .bind(CustomerDTO::getYearOfBirth, CustomerDTO::setYearOfBirth);
```

Validation

VAADIN 8

Validation with Binding

```
TextField firstName = ...
TextField lastName = ...

CustomerDTO customer = ...

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);
```

VAADIN 8

Validation with Binding

```
TextField firstName = …
TextField lastName = …

CustomerDTO customer = …

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(firstName)
```

VADIN 8

Validation with Binding

```
TextField firstName = …
TextField lastName = …

CustomerDTO customer = …

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(firstName)
    .withValidator(Validator::notEmpty, "Mandatory field")
```

vaadin}>

VAADIN 8

Validation with Binding

```
TextField firstName = …
TextField lastName = …

CustomerDTO customer = …

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(firstName)
    .withValidator(Validator::notEmpty, "Mandatory field")
    .withValidator(value -> value.length() < 10, "Must be less than 10 chars")
```

VAADIN 8

Validation with Binding

```
TextField firstName = …
TextField lastName = …

CustomerDTO customer = …

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(firstName)
    .withValidator(Validator::notEmpty, "Mandatory field")
    .withValidator(value -> value.length() < 10, "Must be less than 10 chars")
    .bind(CustomerDTO::getFirstname, CustomerDTO::setFirstname);
```

vaadin}>

VAADIN 8

Validation with Binding

```
TextField firstName = ...
TextField lastName = ...

CustomerDTO customer = ...

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(firstName)
    .withValidator(Validator::notEmpty, "Mandatory field")
    .withValidator(value -> value.length() < 10, "Must be less than 10 chars")
    .bind(CustomerDTO::getFirstname, CustomerDTO::setFirstname);

Button save = new Button("Save", e -> binder.writeBeanIfValid(customer));
```

Validation with Conversion

VAADIN 8

Validation with Conversion

```
TextField yearOfBirth = …
Customer customer = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);
```

vaadin}>

VADIN 8

Validation with Conversion

```
TextField yearOfBirth = …
Customer customer = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(yearOfBirth)
```

VAADIN 8

Validation with Conversion

```
TextField yearOfBirth = …
Customer customer = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(yearOfBirth)
    .withValidator(value -> value.length() == 4, "Must have 4 characters")
```

VAADIN 8

Validation with Conversion

```
TextField yearOfBirth = …
Customer customer = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(yearOfBirth)
    .withValidator(value -> value.length() == 4, "Must have 4 characters")
    .withConverter(Integer::valueOf, String::valueOf)
```

VAADIN 8

Validation with Conversion

```
TextField yearOfBirth = …
Customer customer = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(yearOfBirth)
    .withValidator(value -> value.length() == 4, "Must have 4 characters")
    .withConverter(Integer::valueOf, String::valueOf)
    .withValidator(value -> value < 2000, "Must be before year 2000")
```

vaadin}>

VAADIN 8

Validation with Conversion

```
TextField yearOfBirth = …
Customer customer = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(yearOfBirth)
    .withValidator(value -> value.length() == 4, "Must have 4 characters")
    .withConverter(Integer::valueOf, String::valueOf)
    .withValidator(value -> value < 2000, "Must be before year 2000")
    .bind(CustomerDTO::getYearOfBirth, CustomerDTO::setYearOfBirth);
```

VAADIN 8

Validation with Conversion

```
TextField yearOfBirth = …
Customer customer = …

customer.setYearOfBirth(1984);

Binder<CustomerDTO> binder = new Binder<>(CustomerDTO.class);

binder.forField(yearOfBirth)
    .withValidator(value -> value.length() == 4, "Must have 4 characters")
    .withConverter(Integer::valueOf, String::valueOf)
    .withValidator(value -> value < 2000, "Must be before year 2000")
    .bind(CustomerDTO::getYearOfBirth, CustomerDTO::setYearOfBirth);

Button save = new Button("Save", e -> {
  try {
    binder.writeBean(customer);
  } catch (ValidationException ve) {
    ve.getFieldValidationErrors().forEach(error -> …);
  }
});
```

Session’s content

- The Good Ol’ Boy - Vaadin 7
- The Brand New Field Model of Vaadin 8
- **CustomField<T>**
- Where are my Items & Properties?
- Being Lazy & Happy without Containers

c.v.ui.CustomField<T>

c.v.ui.CustomField<T>

For making Fields for your business types

CUSTOMFIELD

```
public class MoneyField extends CustomField<Money> {
```

CUSTOMFIELD

```
public class MoneyField extends CustomField<Money> {

  private TextField textField;
  private Label currencyCode;
```

CUSTOMFIELD

```
public class MoneyField extends CustomField<Money> {

  private TextField textField;
  private Label currencyCode;

  private BigDecimal parseValue() throws ParseException {
    return  …;
  }

  private String formatValue(BigDecimal parsedValue) {
    return …;
  }
```

CUSTOMFIELD

```
public class MoneyField extends CustomField<Money> {

 private TextField textField;
 private Label currencyCode;

 private BigDecimal parseValue() throws ParseException {
  return  …;
 }

 private String formatValue(BigDecimal parsedValue) {
  return …;
 }

   @Override
 protected Component initContent() {
  return new HorizontalLayout(currencyCode, textField);
 }
```

```
  @Override
public Money getValue() {
  if (StringUtils.isEmpty(textField.getValue())) {
    return null;
  }

  try {
    return Money.of(currencyCode.getValue(), parseValue());
  } catch (ParseException e) {
    textField.setComponentError(new UserError("Invalid format”));
    return null;
  }
}

  @Override
protected void doSetValue(Money value) {
  if (value == null) {
    textField.clear();
    currencyCode.setValue(null);
  } else {
    textField.setValue(formatValue(value.getAmount()));
    currencyCode.setValue(value.getCurrencyCode());
  }
}
```

Session’s content

- The Good Ol’ Boy - Vaadin 7
- The Brand New Field Model of Vaadin 8
- CustomField<T>
- **Where are my Items & Properties?**
- Being Lazy & Happy without Containers

c.v.data.Item

TO

<T>

ComboBox

TO

ComboBox<Country>

```
ountry c = (Country) comboBox.getValue(
```

TO

```
Country c = comboBox.getValue()
```

How would you make
a dropdown of countries?

VAADIN 7

```
BeanItemContainer<Country> container =
     new BeanItemContainer<>(Country.class);
```

```
BeanItemContainer<Country> container =
    new BeanItemContainer<>(Country.class);

container.addAll(getCountries());
```

```
BeanItemContainer<Country> container =
    new BeanItemContainer<>(Country.class);

container.addAll(getCountries());

ComboBox countrySelector = new ComboBox();
```

```
BeanItemContainer<Country> container =
    new BeanItemContainer<>(Country.class);

container.addAll(getCountries());

ComboBox countrySelector = new ComboBox(); countrySelector.setContainerDataSource(container);
```

```
BeanItemContainer<Country> container =
    new BeanItemContainer<>(Country.class);

container.addAll(getCountries());

ComboBox countrySelector = new ComboBox();
countrySelector.setContainerDataSource(container);
countrySelector.setItemCaptionMode(ItemCaptionMode.PROPERTY);
```

```
BeanItemContainer<Country> container =
    new BeanItemContainer<>(Country.class);

container.addAll(getCountries());

ComboBox countrySelector = new ComboBox(); countrySelector.setContainerDataSource(container);
countrySelector.setItemCaptionMode(ItemCaptionMode.PROPERTY);
countrySelector.setItemCaptionPropertyId("name");
```

vaadin}>

How would you make
a dropdown of countries?

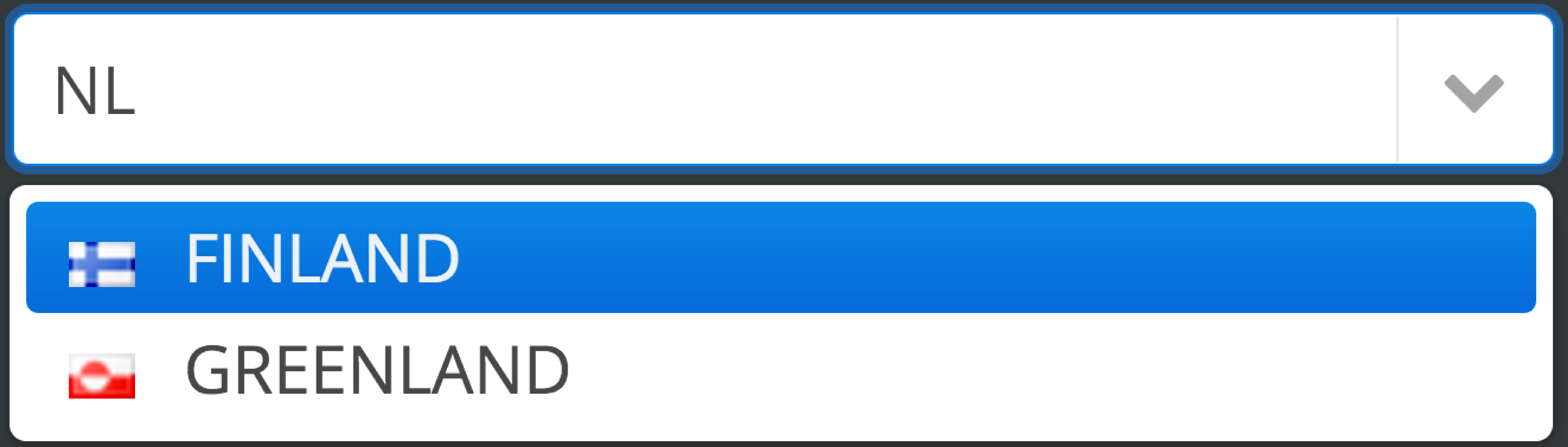
VAADIN 8

```
ComboBox<Country> countrySelector = new ComboBox<>();
countrySelector.setItems(getCountries());
```

```
ComboBox<Country> countrySelector = new ComboBox<>();
countrySelector.setItems(getCountries());
countrySelector.setItemCaptionGenerator(Country::getName);
```

How would you filter entries in a dropdown?

VAADIN 7


NL

```
countrySelector.setFilteringMode(FilteringMode.CONTAINS);
```

OR

```
countrySelector.setFilteringMode(FilteringMode.STARTSWITH);
```

How would you filter entries in a dropdown?

VAADIN 8

```
ListDataProvider<Country> dataProvider = DataProvider.fromStream(getCountries().stream().filter(...));
countrySelector.setDataProvider(dataProvider);
countrySelector.setDataProvider(DataProvider<Country, String> dataProvider)
```

```
public interface DataProvider<Country, String> {

   int size(Query<String> t);

   Stream<Country> fetch(Query<String> query);
}
```

DataProvider<Country, String>

DataProvider<Country, String>

```
java.util.function.Predicate<T>

CriteriaQuery.where(...)

Service.findCountries(String)
```

Plugin to any data source lazily!

Query<F>

```
public class Query<T, F> {

  int getOffset();

  int getLimit();

  List<SortOrder<String>> getSortOrders();

  Optional<F> getFilter();
}
```

```
DataProvider<Person, Void> dataProvider = DataProvider.fromCallbacks(
 // First callback fetches items based on a query
 query -> {
  // The index of the first item to load
  int offset = query.getOffset();

  // The number of items to load
  int limit = query.getLimit();

  List<Person> persons = getPersonService().fetchPersons(offset, limit);

  return persons;
 },
 // Second callback fetches the number of items for a query
 query -> getPersonService().getPersonCount()
);
```

The classic Container discrepancy

VAADIN 7

```
Grid grid = new Grid();
grid.addColumn("firstName");
grid.addColumn("lastName");
grid.addColumn("yearsOld");
grid.setContainerDataSource(
    new BeanItemContainer<>(CustomerDTO.class));
```

HTTP Status 500 - com.vaadin.server.ServiceException: java.lang.IllegalStateException:

Found at least one column in Grid that does not exist in the given container: yearsOld with the header “Years Old”.

Call removeAllColumns() before setContainerDataSource() if you want to reconfigure the columns based on the new container.

#FFS

```
Grid grid = new Grid();
grid.addColumn("firstName");
grid.addColumn("lastName");
grid.addColumn("yearsOld");
```

```
Grid grid = new Grid();
grid.addColumn("firstName");
grid.addColumn("lastName");
grid.addColumn("yearsOld");



GeneratedPropertyContainer generatedProps =
  new GeneratedPropertyContainer(
    new BeanItemContainer<>(CustomerDTO.class));
generatedProps.addContainerProperty("yearsOld", Integer.class, null);



grid.setContainerDataSource(generatedProps);
```

```
HTTP Status 500 - com.vaadin.server.ServiceException:  java.lang.UnsupportedOperationException:

GeneratedPropertyContainer does not support adding properties.
```

#FFS

THIS ISN’T FUN ANYMORE…

```
Grid grid = new Grid();
grid.addColumn("firstName");
grid.addColumn("lastName");
grid.addColumn("yearsOld");

GeneratedPropertyContainer generatedProps =
  new GeneratedPropertyContainer(
    new BeanItemContainer<>(CustomerDTO.class));




grid.setContainerDataSource(generatedProps);
```

vaadin}>

```
generatedProps.addGeneratedProperty("yearsOld", new
    PropertyValueGenerator<Integer>() {

        @Override
        public Class<Integer> getType() {
            return Integer.class;
        }

        @Override
        public Integer getValue(Item item, Object itemId,
             Object propertyId) {

             return LocalDate.now().getYear() -
                 ((CustomerDTO)itemId).getYearOfBirth();
        }
    });
```

#FFS

OH REALLY!?

Bye Bye Containers;
we've had enough!

VAADIN 8

vaadin}>

```
Grid<CustomerDTO> grid = new Grid<>();
```

```java
Grid<CustomerDTO> grid = new Grid<>();
grid.addColumn("firstName", CustomerDTO::getFirstname);
```

```java
Grid<CustomerDTO> grid = new Grid<>();
grid.addColumn("firstName", CustomerDTO::getFirstname);
grid.addColumn("lastName", CustomerDTO::getLastname);
```

```java
Grid<CustomerDTO> grid = new Grid<>();
grid.addColumn("firstName", CustomerDTO::getFirstname);
grid.addColumn("lastName", CustomerDTO::getLastname);
grid.addColumn("yearsOld",
    customer -> String.valueOf(LocalDate.now().getYear() - customer.getYearOfBirth()));
```

```java
Grid<CustomerDTO> grid = new Grid<>();
grid.addColumn("firstName", CustomerDTO::getFirstname);
grid.addColumn("lastName", CustomerDTO::getLastname);
grid.addColumn("yearsOld",
    customer -> String.valueOf(LocalDate.now().getYear() - customer.getYearOfBirth()));




grid.setDataProvider(DataProvider<CustomerDTO, F> customerDataProvider);
```

Lessons learned

- Vaadin 8 - built on Java 8
- Lambdas and Functional paradigm at large
- DataBinding completely redone
- Drops Containers, Items and Properties
- Targets simplicity and flexibility

T H A N K Y O U !

https://vaadin.com/framework

@haijian_wang