

5.Data Binding Components

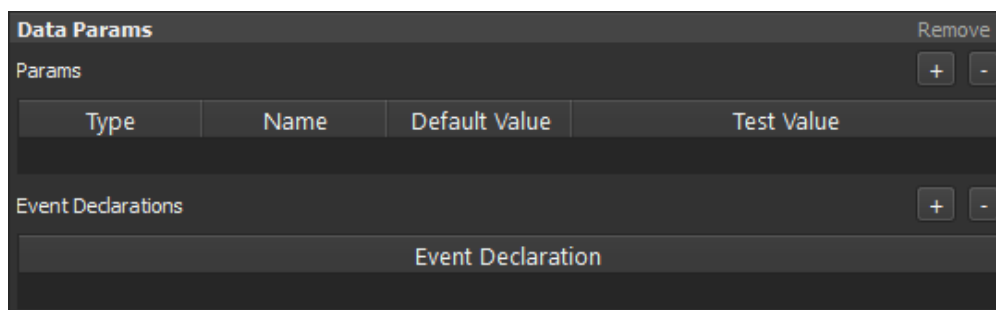
- [Data Params](#)
- [Local Data](#)
- [Child Component](#)
- [Биндинг свойств контрола и компонент](#)
- [Обработка вопросов в Issue Navigator](#)

Data Binding System - позволяет устанавливать соединение между UI (пользовательским интерфейсом) редактора QuickEd и бизнес-логикой.

В данном документе рассмотрены основные компоненты для работы с системой Data Binding.

Data Params

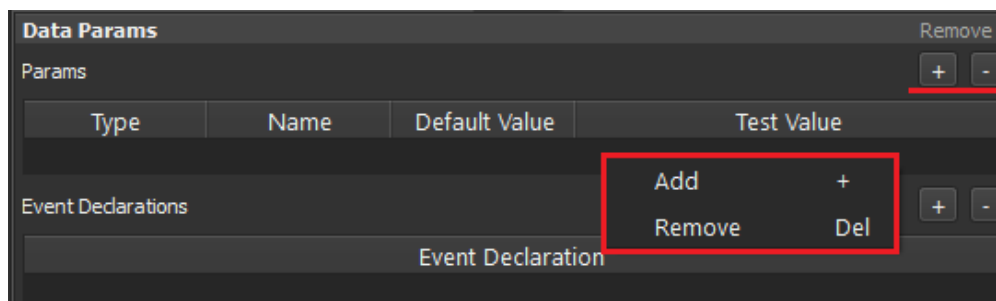
Компонента Data Params служит для задания глобальных параметров, которые будут доступны от рутового контрола всем его дочерним контролам. Компонента имеет следующий вид:



Тут будет рассмотрена работа с блоком Params:

Добавить строку для записи параметра можно тремя способами:

- воспользоваться кнопками +/-;
- вызвать контекстное меню и выбрать команду Add;
- использовать шорткат "+", предварительно установив курсор на пустое поле.



Для удаления, соответственно, способы аналогичные, но со знаком минус.

По умолчанию добавляется параметр типа int:

Data Params
Remove

Params
+ -

| Type | Name | Default Value | Test Value |
|------|------|---------------|------------|
| int | var | 0 | 0 |

Event Declarations
+ -

Event Declaration

Все поля параметра доступны для редактирования.

В поле **Type** вводится тип параметра.

Доступные типы данных:

- int
- uint
- int64
- uint64 //беззнаковый int64
- float
- string
- bool
- vector
- map //ключ простой тип (int, string, ...) и enum
- set
- color
- vector2
- rect

а также сложные типы:

- enum
- type

В поле **Name** указывается имя параметра.

В поле **Default Value**

- Значение **null** - вариант для структур, векторов и map.
- Если это простые типы, то 0. Значения Default Value в редакторе не отображаются.

В поле **Test Value** задается тестовое значение параметра, которое будет отображаться в редакторе.

Data Params
Remove

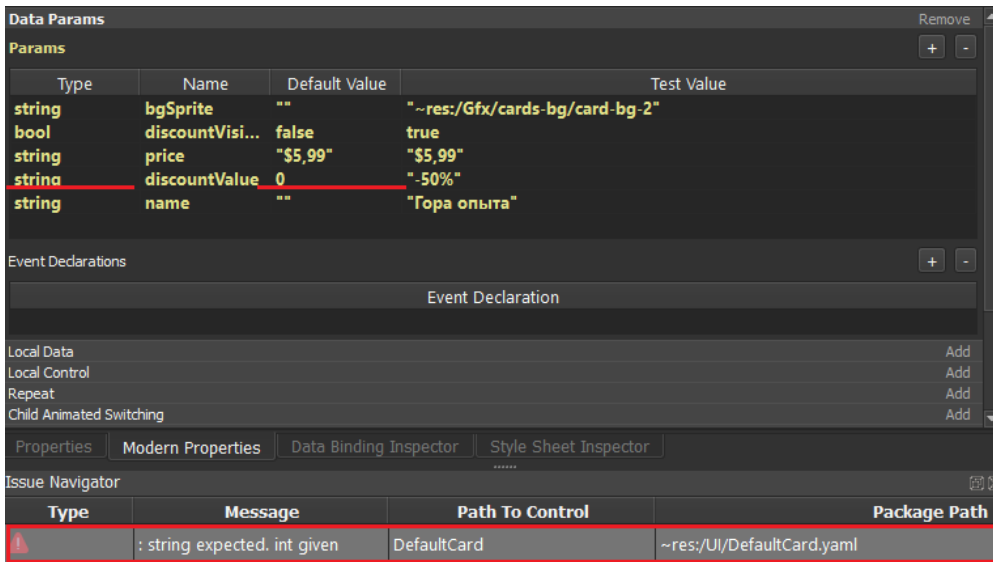
Params
+ -

| Type | Name | Default Value | Test Value |
|--------|-----------------|---------------|---------------------------------|
| string | bgSprite | "" | "--res:/Gfx/cards-bg/card-bg-2" |
| bool | discountVisi... | false | true |
| string | price | "\$5,99" | "\$5,99" |
| string | discountValue | 0 | "-50%" |
| string | name | "" | "Гора опыта" |

Event Declarations
+ -

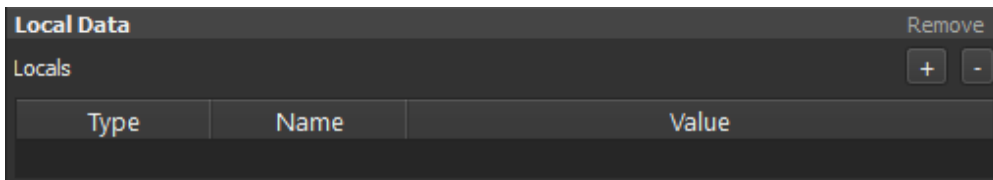
Event Declaration

В случае, если в поле Default Value либо Test Value задан тип данных отличный от того, что задано в поле Type, на панели Issue Navigator будет выдаваться сообщение об ошибке.

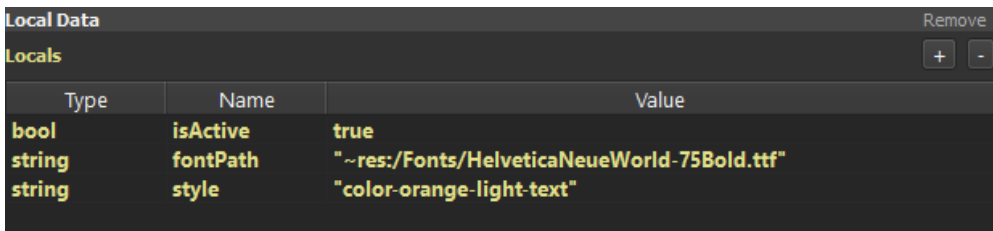


Local Data

Компонента Local Data необходима для задания локальных аргументов, которые будут доступны только в рамках одного yaml-файла. Компонента имеет следующий вид:



Структура практически такая же как и у компоненты Data Params, за исключением поля Value. В компоненте отсутствуют поля Default Value и Test Value вместо них одно поле Value.



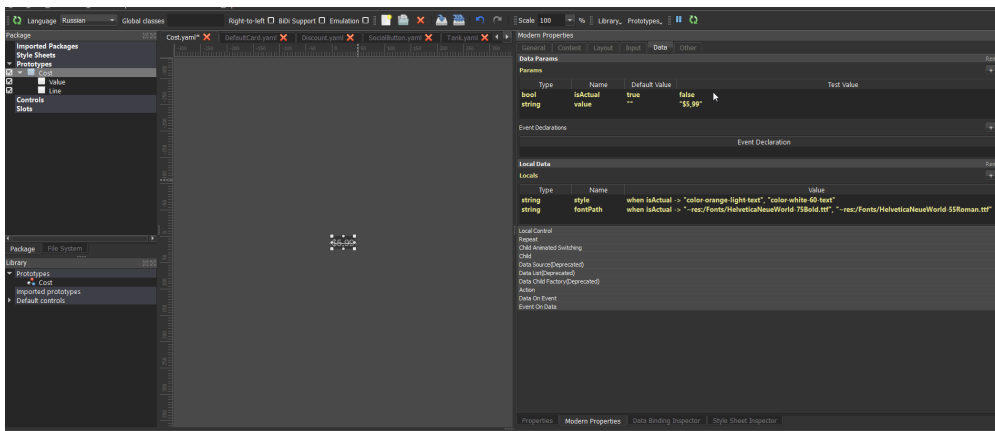
Поле Value доступно не только для ввода значения, а также для работы с функциями и операторами, например с условным оператором when:

| | |
|------|--|
| when | when <condition> <value> |
| | Например: |
| | when c == 1 -> "one", c == 4 -> "four", "?". |

Рассмотрим работу условного оператора when на примере работы Data Params и Local Data:

| Data Params | | | | Remove | |
|-------------------|----------|--|------------|--------|--------|
| Params | | | | + - | |
| Type | Name | Default Value | Test Value | | |
| bool | isActual | true | false | | |
| string | value | "" | "\$5,99" | | |
| Event Dedarations | | | | | + - |
| Event Declaration | | | | | |
| Local Data | | | | | Remove |
| Locals | | | | | + - |
| Type | Name | Value | | | |
| string | style | when isActual -> "color-orange-light-text", "color-white-60-text" | | | |
| string | fontPath | when isActual -> "~res:/Fonts/HelveticaNeueWorld-75Bold.ttf", "~res:/Fonts/HelveticaNeueWorld-55Roman.ttf" | | | |

В компоненте Data Params задан параметр isActual с типом данных bool. А в компоненте Local Data задано два аргумента с условными оператором, условие завязано на параметре isActual. В первом аргументе если isActual равно true, то будет подтягиваться стиль color-orange-light-text, если isActual равно false, то будет подтягиваться стиль color-white-60-text, соответственно. У второго аргумента ситуация аналогично, но касается шрифта. Получаем следующий результат:



Child Component

Child компонента имеет следующий вид:

| Child [0] | kOldExpression |
|----------------|----------------|
| Type | kPrototype |
| If | kControl |
| Prototype Path | kRepeat |
| Args | |
| Event Actions | |

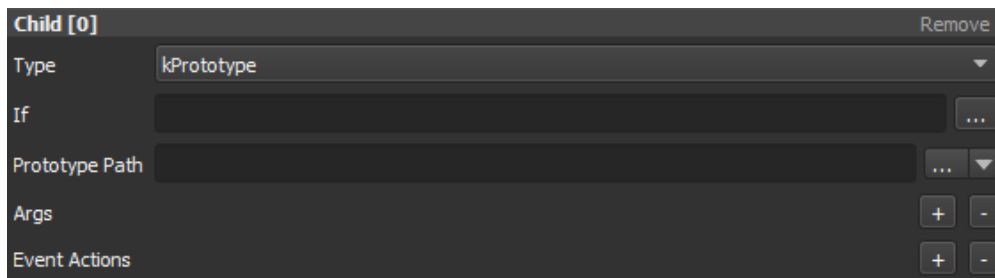
В поле **Type** присутствует четыре типа представления: OldExpression, Prototype, Control, Repeat. Каждому типу соответствует свой набор дополнительных свойств. Свойства будут меняться в зависимости от выбранного типа.

If - условие срабатывания, если у контрола есть несколько child компонент, то сработает только первая, у которой условие будет равно true.

Не заданное условие - равноценно true.

Также стоит иметь ввиду, что контрол не будет пересоздаваться, если не поменялось условие или исходные данные для построения контрола.

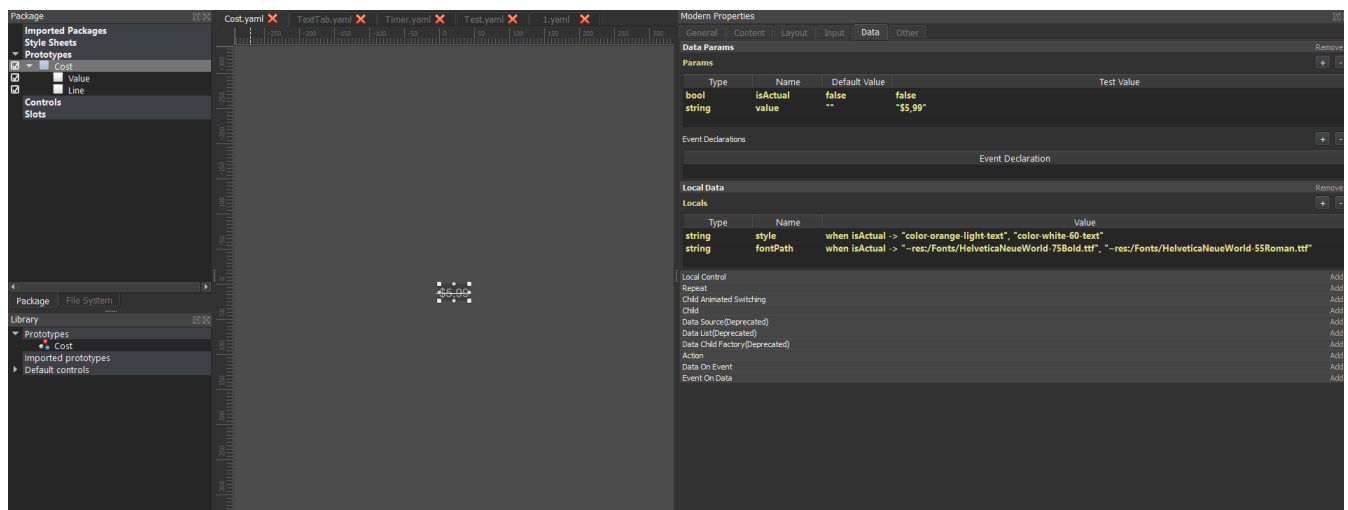
Type - Prototype:



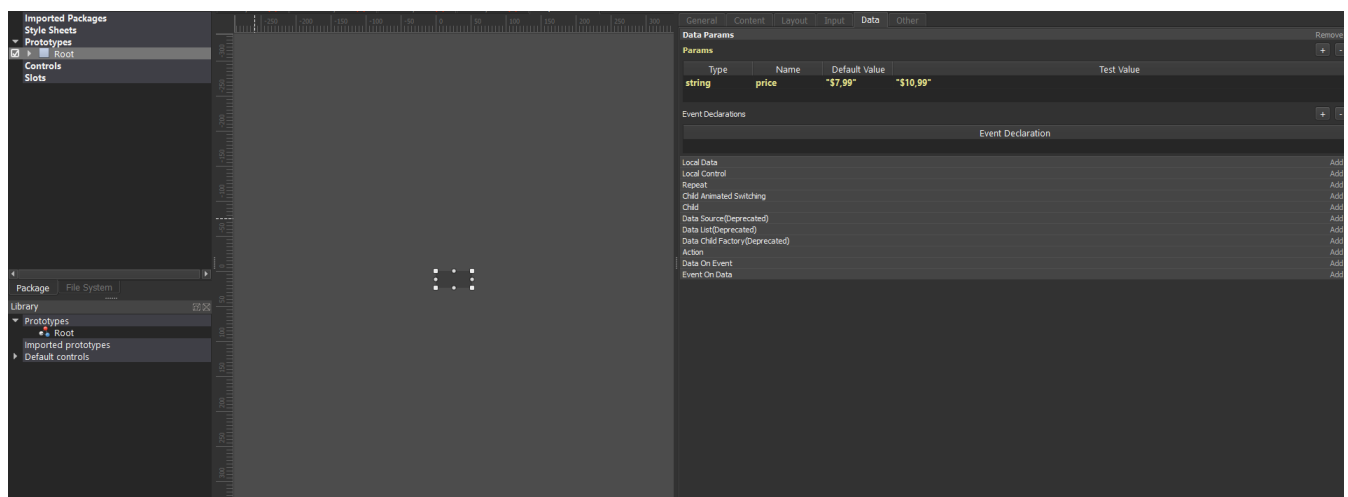
- If - условие срабатывания
- Prototype Path - указывается путь к yamI-файлу (строка)
- Args - задаются аргументы
- Event Actions - задаются эшкены

Пример работы Child - компоненты с Type - Prototype

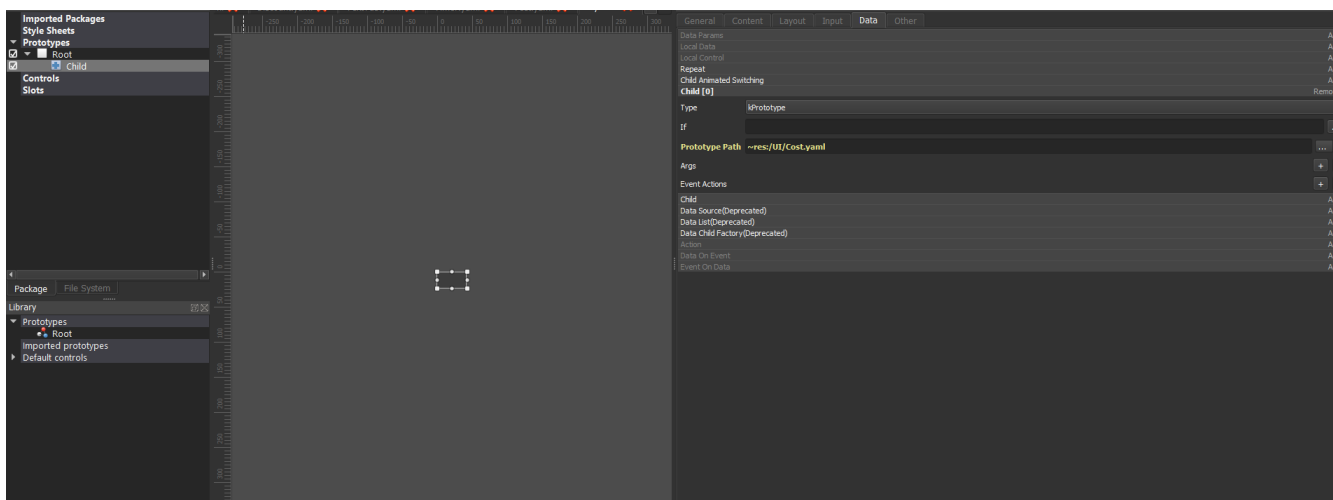
Рассмотрим работу Child компоненты с Type - Prototype на примере прототипа:



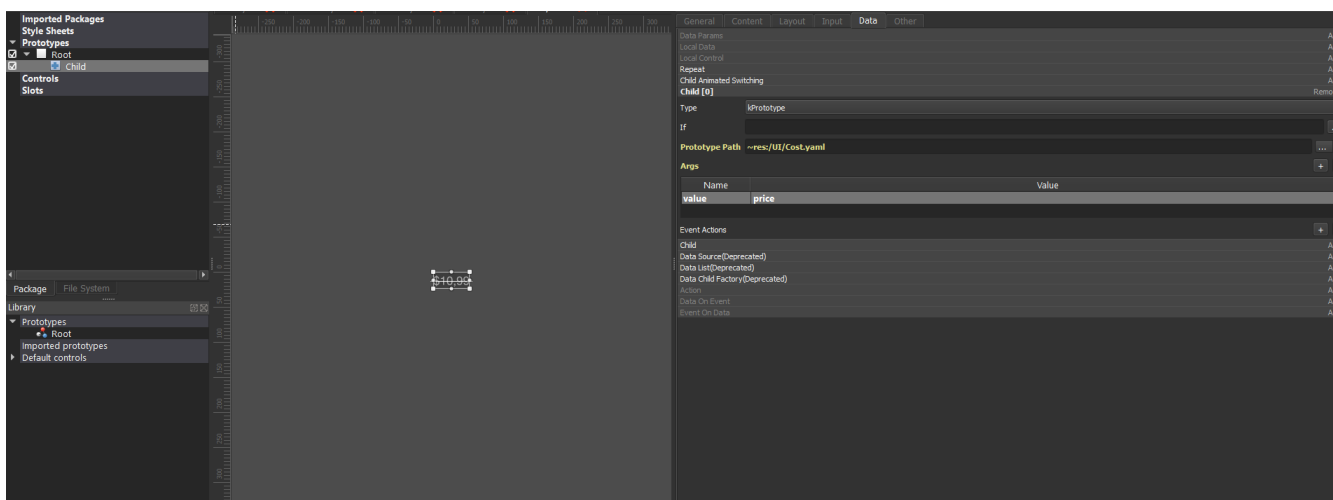
Создадим еще один файл с рутовым контролем в секции Prototype и добавим ему компоненту Data Params с типом данных string:



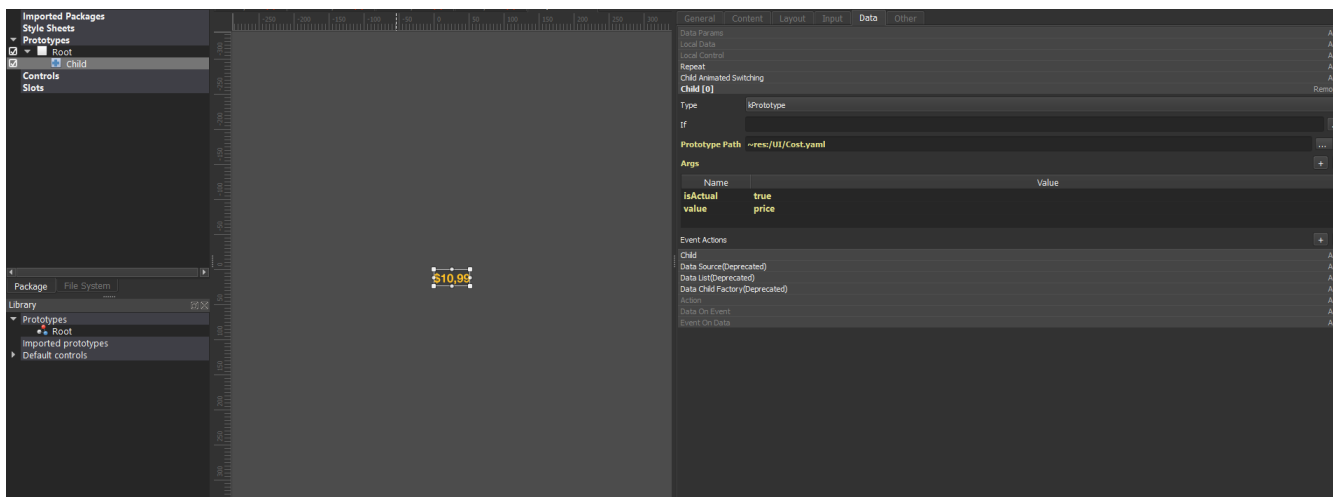
Далее рутовому контролю добавляем дочерний контрол и этому контролю добавляем компоненту Child с типом Prototype. И в поле Prototype Path указываем путь к контролу с прототипом.



Далее переопределяем аргумент в компоненте Child. Изначально в прототипе в файле [~res:/UI/Cost.yaml](#) был задан аргумент value, который имел значение 5.99, а у рутового контрола в данном файле был задан параметр price, у которого тестовое значение было указано как 10.99, таким образом, в поле args мы присвоили аргументу с именем value значение параметра с именем price



Аналогичным образом мы можем переопределить значение isActual с false на true, чтобы изменить стиль и шрифт при отображении текста.

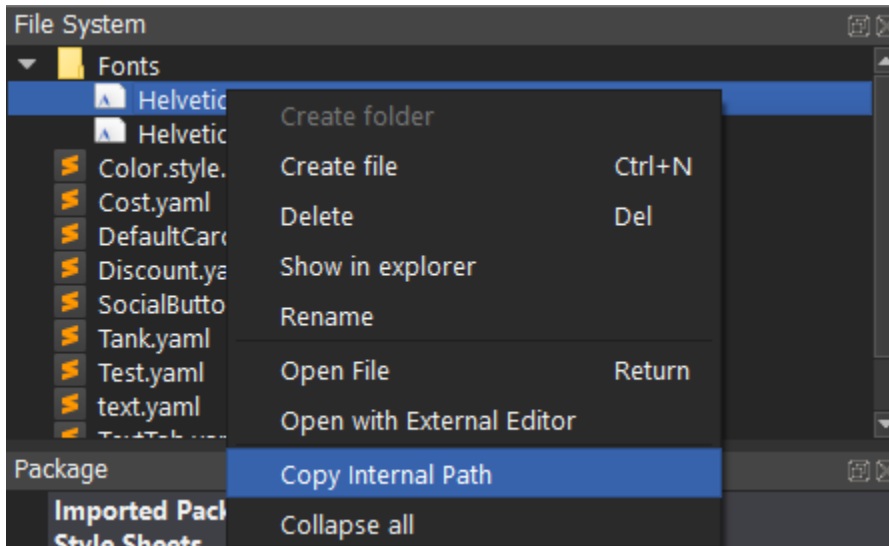


Биндинг свойств контрола и компонент

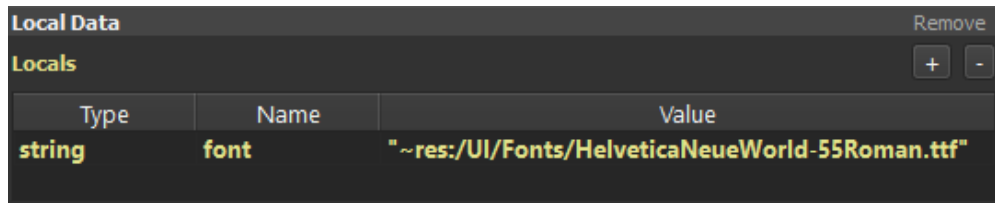
Для биндинга доступны некоторые свойства UIControl'a и некоторые свойства компонент.

В текстовой компоненте существует возможность биндить свойство Font Path. Для это в Data Params либо Local Data создадим переменную тип string и в кавычках укажем путь к шрифту.

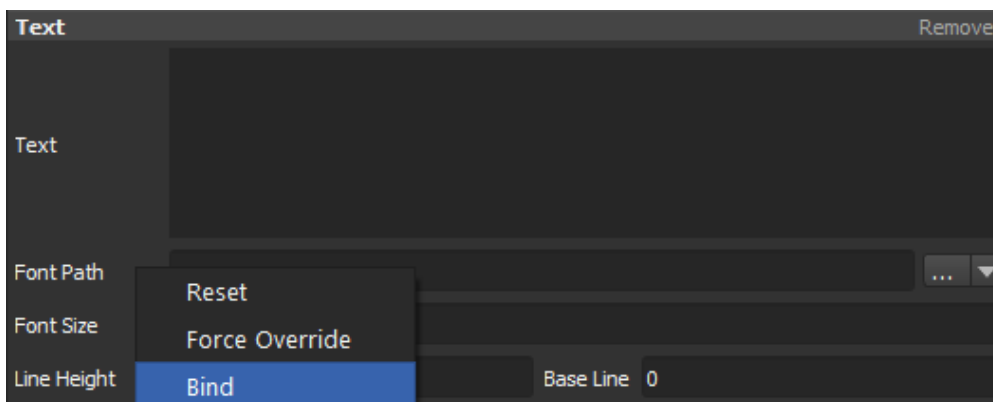
Путь к шрифту можно узнать путем перехода на панель File System, раскрыть папку Fonts и, кликнув правой кнопкой мыши по файлу со шрифтом, выбрать пункт Copy Internal Path.



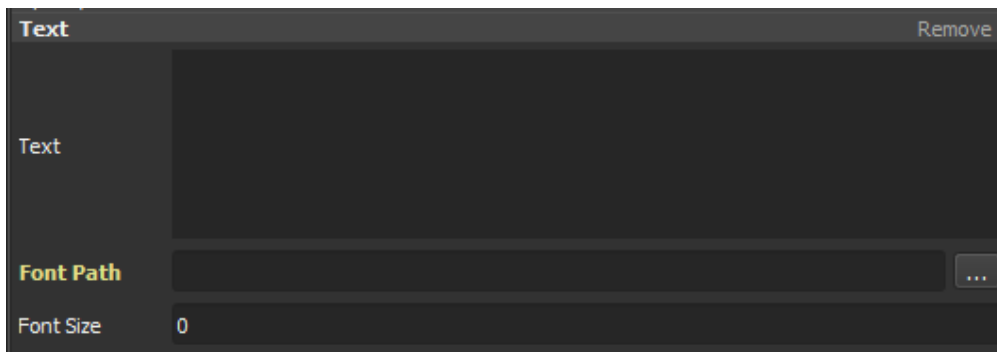
Затем вставить его в поле value компоненты Local Data. Получаем следующий результат:



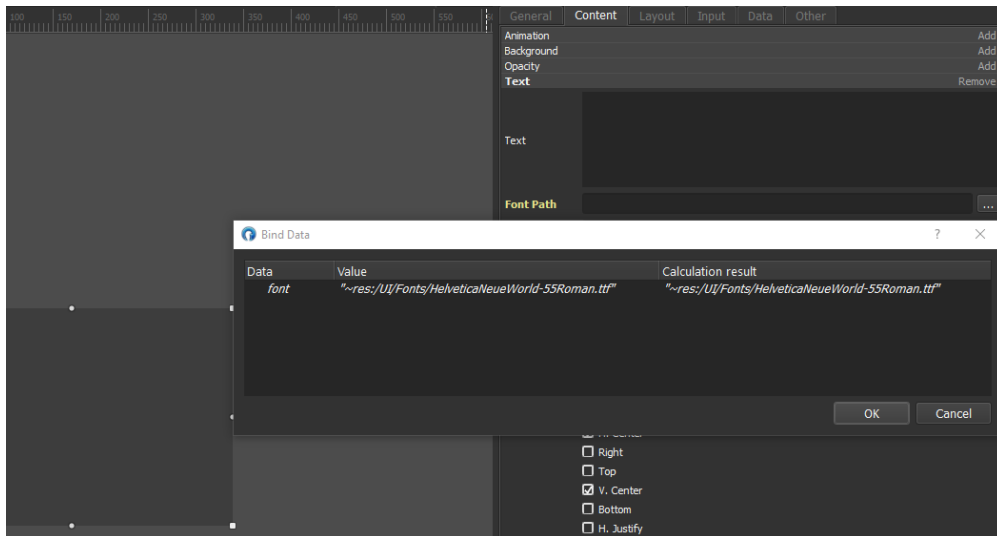
Далее переходим к компоненте Text, к полю Font Path. На панели Modern Properties рядом с надписью Font Path кликаем правой кнопкой мыши и выбирает пункт Bind.



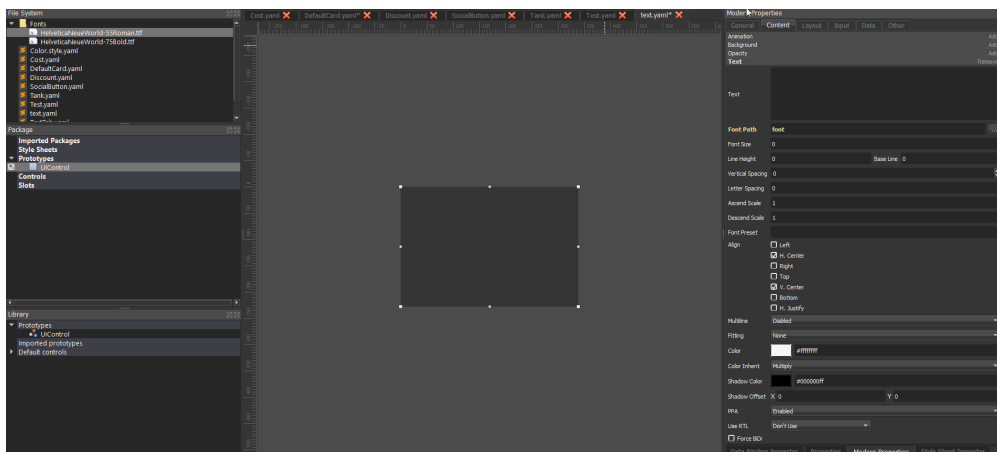
Справа появится кнопка троеточия.



Откроется окно Bind Data, в котором будет доступен созданным нами аргумент.

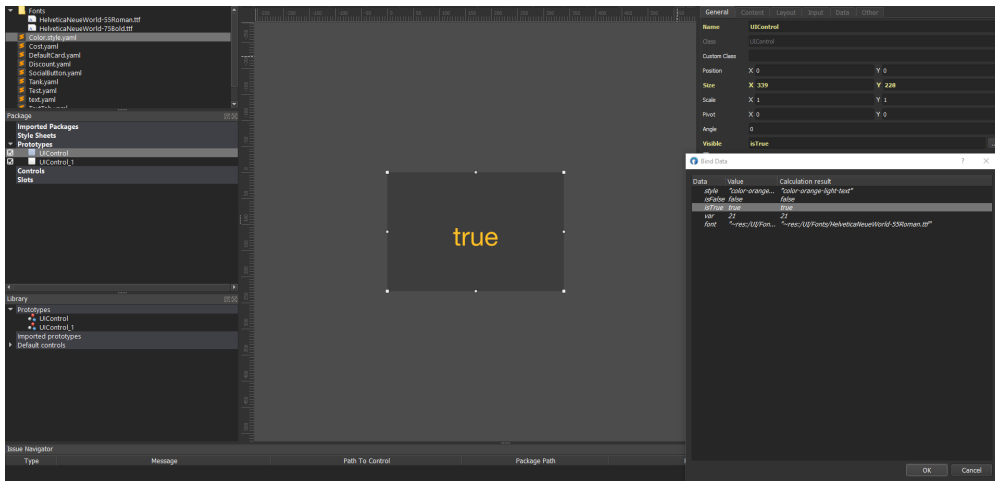


Поле Text биндится аналогично, но в данном поле можно указывать параметры и аргументы различных типов.

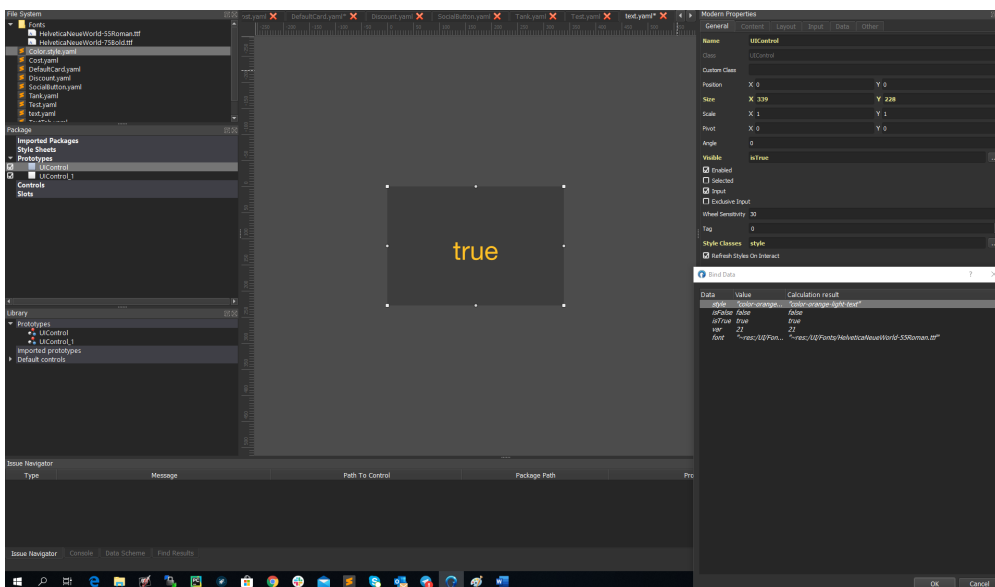


Принцип биндинга свойств контролов и свойств компонент аналогичен приведенному примеру с компонентой Text.

Биндинг свойства visible у UIControl'a:



Биндинг свойства Style Classes у UIControl'a:



Обработка вопросов в Issue Navigator

Следует иметь ввиду, что если изначально был выбран один тип child - компоненты и заданы свойства, уникальные для этого типа, то после смены типа, эти свойства не сбрасываются автоматически, их необходимо сбросить самостоятельно. При этом, если не сброшено значение проперти, которая не соответствует данному типу child-компоненты, на панели Issue Navigator появится сообщение об ошибке.

Child [0]

Remove

Type

kControl

If

...

Prototype Path

~res:/UI/CHILD/slots.yaml

...

Control

...

Child

Add

Data Source(Deprecated)

Add

Data List(Deprecated)

Add

Data Child Factory(Deprecated)

Add

Action

Add

Data On Event

Add

Event On Data

Add

Properties

Modern Properties

Style Sheet Inspector

Issue Navigator

| Type | Message | Path To Control | Package Path | Property Name |
|------|--|-----------------|------------------|-------------------|
| | You should reset property 'Prototype Path' when 'Type' is 'kControl' | Control | ~res:/UI/new_... | Child[0]/Proto... |

Также на панели Issue Navigator появится сообщение об ошибке, если для типа Control в поле Control результат выражение не является объект типа control:

Child [0]

Remove

Type

kControl

If

...

Control

a + b

...

Child

Add

Data Source(Deprecated)

Add

Data List(Deprecated)

Add

Properties

Modern Properties

Style Sheet Inspector

Issue Navigator

| Type | Message | Path To Control | Package Path | Property Name |
|------|--------------------------------------|-----------------|------------------|------------------|
| | Result of expression must be control | Control | ~res:/UI/new_... | Child[0]/Control |