Системные действия

В Versium предусмотрены два вида системных действий:

- Скрипты — серверный .NET-код без взаимодействия с пользователем.
- Визарды (мастера) — интерактивные пошаговые формы с пользовательским интерфейсом и произвольной бизнес-логикой.

Встроенный редактор кода

- Для написания системных действий, в Versium есть встроенный редактор кода со множеством сниппетов и автодополнений.
- Редактор кода подсказывает с учетом текущего контекста мастера (какие элементы уже были добавлены на старницу), а также с учетом контекста модели базы данныз (какие существуют таблицы в бд и какой в них набор полей).
- Мы старались максимально упростить жизнь разработчиков, но данный раздел требуется для понимания общей структуры и возможностей данного модуля.

Ниже приведено описание, когда и как использовать каждый вид, а также базовые принципы разметки и логики мастеров.

Когда использовать скрипт, а когда мастер

- Используйте скрипты, если:
  - Взаимодействие с пользователем не требуется.
  - Действие должно выполняться автоматически (по событию, по расписанию, по окончании интеграции).
  - Требуется быстрая серверная операция из UI по нажатию кнопки без ввода дополнительных данных.
  - Примеры: отправка письма в фоне, перерасчет агрегатов после интеграции, перевод инцидента в статус «В работе».
- Используйте визарды (мастера), если:
  - Нужен пошаговый ввод данных пользователем с валидацией.
  - Необходим UI: поля ввода, таблицы, кнопки, виджеты, проверка обязательных полей.
  - Требуется логика любой сложности с разветвлениями и контролем переходов между шагами.

Скрипты

Скрипт — это .NET-код, который выполняется без UI. Его можно:

- Запускать в фоне по системным событиям.
- Привязывать к кнопкам на формах объектов (быстрые операции).
- Вызывать по завершению интеграций, Cron/планировщикам и т. п.

Минимальный каркас скрипта:

```
// Дописать в запись, в контексте которой выполняется скрипт, строку в поле Комментарий
contextData["Comment"] = (string)contextData["Comment"] + $"\nChangedFromScript {DateTime.UtcNow:yy-MM-dd HH:mm:ss}";
// Дописать в ссылочное поле Ответственный Id сотрудника, связанного с пользователем, запустившим скрипт
contextData["Responsible"] = user.EmployeeId;
```

Визарды (мастера)

Мастер — это смесь:

- XML-разметки UI (поля, таблицы, кнопки, сетка) и
- .NET-кода, который выполняется:
  - при открытии мастера,
  - при нажатии на кнопки,
  - при переходах между страницами,
  - при завершении мастера (по кнопке «Готово»).

Мастера обычно многостраничные (паттерн «Далее, Далее, Готово») и ведут пользователя по шагам с валидацией.

XML-разметка мастера

- Элемент верхнего уровня: `wizard`
- Внутри него вложены страницы: `page`
- Внутри страницы разметка построчная: каждая строка — элемент `row`.

- Внутри `row` размещаются UI-элементы `element` разных типов (например, `type="string"` , `type="int"` , `type="button"` ,...).
- Для каждого UI-элемента обязательны атрибуты:
  - `name` — внутреннее имя (идентификатор элемента),
  - `title` — заголовок/лейбл,
  - `width` — ширина в колонках сетки (от 1 до 12).
- Сетка 12-колоночная: сумма `width` элементов в строке не должна превышать 12.

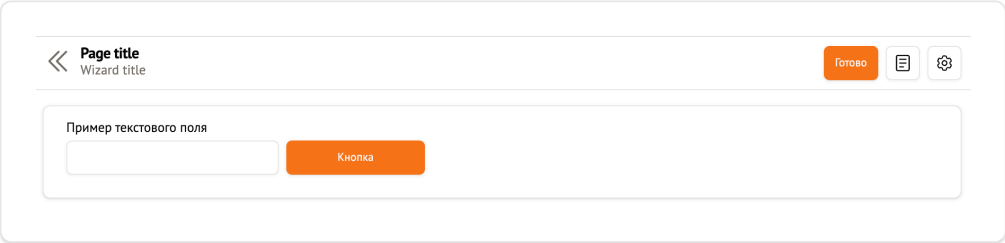
Простейший пример мастера

```
<wizard title="Wizard title" endAction="ToContext">
  <page name="page1" title="Page title">
    <onLoadScript>
      <![CDATA[
        // Скрипт, который выполнится при открытии мастера
      ]]>
    </onLoadScript>

    <row>
      <element name="ExampleStringInput" title="Пример текстового поля" type="string" width="3" value="" />
      <element name="ExampleButton" title="Кнопка" type="button" width="2">
        <![CDATA[
          // Скрипт, который выполнится по нажатию на кнопку
        ]]>
      </element>
    </row>

    <transition toPage="FINISH" title="Готово">
      <validationScript>
        <![CDATA[
          // Скрипт валидации перехода
        ]]>
      </validationScript>
      <transitionScript>
        <![CDATA[
          // Скрипт, который выполнится при нажатии на кнопку Готово (если будет пройдена валидация)
        ]]>
      </transitionScript>
    </transition>
  </page>
</wizard>
```

Как выглядит результат рендера



Пример раскладки по сетке

```
<row>
  <element name="Name" type="string" title="Имя" width="6" />
  <element name="Surname" type="string" title="Фамилия" width="3" />
  <element name="Age" type="int" title="Возраст" width="3" />
</row>
```

Рекомендации по проектированию

- Разделяйте ответственность: UI и валидации — в мастере, тяжелая серверная логика и интеграции — в скриптах/сервисах.
- Делите мастера на логические шаги с четкими валидаторами.
- Изменяйте данные в базе данных только внутри скрипта FINISH в мастере.
- Соблюдайте сетку 12 колонок; не переполняйте строки.

- Обеспечивайте понятные сообщения об ошибках и результаты действий для пользователя.