## Тема «УПРАВЛЯЕМАЯ ФОРМА. ОБРАБОТЧИКИ 1С 8.3»

**Цель**
Познакомиться с управляемыми формами.Обработчики.

### Введение

## ОБРАБОТЧИКИ СОБЫТИЙ 1С:ПРЕДПРИЯТИЕ 8.3

Особенностью обработки событий среде 1С:Предприятия 8 является то, что имя процедуры-обработчика в одних случаях должно совпадать с именем события, а в других случаях может от него отличаться. ссылка на ИТС из прошлого урока дополнительная литература

Процедуры-обработчики событий //its.1c.ru

Жизненный цикл управляемой формы. Шпаргалка разработчика ссылка Станислав Ганиев@infostart.ru Jun 2018

### Форма

### События жизненного цикла формы

Открытие объекта в форме

| МОДУЛЬ ФОРМЫ &НАКЛИЕНТЕ | МОДУЛЬ ФОРМЫ &НАСЕРВЕРЕ | МОДУЛЬ ОБЪЕКТА &НАСЕРВЕРЕ | |
|---|---|---|---|
| | • **ПриЧтенииНаСервере (<ТекущийОбъект>)** — выполняется только для существующего объекта, подготовка доп.данных, зависящих от данных объекта. Объект — основной элемент открываемой формы;<br>  • ТекущийОбъект — объект, прочитанный из БД (доступны экспортные методы объекта)<br>• **ОбработкаЗаполнения (<ДанныеЗаполнения>, <ТекстЗаполнения>, <СтандартнаяОбработка>)** — Только для нового объекта. Начальное заполнение объекта данными. Доступны<br>  • ДанныеЗаполнения (ссылка на объект-основание либо структура отбора). В случае<br>  • СтандартнаяОбработка = Истина, после обработки заполнения, обрабатываются сначала ДанныеЗаполнения, затем ЗначенияЗаполнения | | |
| | | • **ПриСозданииНаСервере (<Отказ>, <СтандартнаяОбработка>)** — Окончательная подготовка формы (ее представления) к открытию<br>  •<br>  • форма, открытая для создания нового объекта, получает новый пустой несохраненный **Объект**, для которого<br>  метод **Объект.ЭтоНовый()** возвращает **Истина**<br>  • фактически этот метод приводит к ошибке «Метод объекта не обнаружен (ЭтоНовый)» и следует<br>  использовать **ЗначениеЗаполнено(Объект.Ссы** | |

лка)

- **ПриОткрытии (<Отказ>)** — Действия, связанные с открытием, которые на сервере выполнить невозможно; выполняемые тогда, когда форма наверняка открывается. Последний обработчик перед открытием, в котором можно отказаться от открытия

## Запись объекта в форме

| МОДУЛЬ ФОРМЫ &НАКЛИЕНТЕ | МОДУЛЬ ФОРМЫ &НАСЕРВЕРЕ | МОДУЛЬ ОБЪЕКТА &НАСЕРВЕРЕ |
|---|---|---|
| • **ПередЗаписью(<Отказ>)** — Анализ готовности вспомогательных данных для записи объекта | | |
| | • **ОбработкаПроверкиЗаполненияНаСервере(<Отказ>, <ПроверяемыеРеквизиты>)** — Проверка данных, не относящихся к объекту.<br>  • ПроверяемыеРеквизиты содержит массив имен реквизитов формы | |
| | | • **ОбработкаПроверкиЗаполнения(<Отказ>, <ПроверяемыеРеквизиты>)** — Всё, что относится к проверкам реквизитов основного объекта БД.<br>  • ПроверяемыеРеквизиты содержит массив имен реквизитов объекта |
| | **НАЧАЛО ТРАНЗАКЦИИ** | |
| | • **ПередЗаписьюНаСервере(Отказ, ТекущийОбъект, ПараметрыЗаписи)** — Объект — основной реквизит формы объекта;<br>  • ТекущийОбъект — объект, который реально будет записан в БД. Анализировать данные и дозаполнять реквизиты нужно через ТекущийОбъект, модификация Объекта ни к чему не приведет. Обработчик вызывается только при записи из формы | |
| | | • **ПередЗаписью(<Отказ>)** — Вызывается при любом способе записи. Данные объекта записываются в БД, но транзакция не закрывается<br>• **ПриЗаписи(<Отказ>)** — Выполняются действия над доп.данными, которые неразрывно связаны с основными данными объекта |
| | **КОНЕЦ ТРАНЗАКЦИИ** | |
| | • **ПриЗаписиНаСервере(<Отказ>, <ТекущийОбъект>, <ПараметрыЗаписи>)** — Предназначение аналогично ПриЗаписи(), но при обработке доп.данных можно использовать данные формы.<br>  • ТекущийОбъект — данные, которые были записаны в БД, работать следует | |

именно с ним; Объект — данные основного реквизита формы, которые были до записи, его модификация бесполезна. Если это запись нового объекта, то у Объект свойство Ссылка пустое, а у ТекущийОбъект уже заполнено

- **ПослеЗаписи(<ПараметрыЗаписи>)** — Действия, которые невозможно выполнить на сервере, или требующие интерактивного взаимодействия с пользователем. Объект гарантированно записан

### Закрытие формы

| МОДУЛЬ ФОРМЫ &НАКЛИЕНТЕ | МОДУЛЬ ФОРМЫ &НАСЕРВЕРЕ | МОДУЛЬ ОБЪЕКТА &НАСЕРВЕРЕ | |
|---|---|---|---|
| • **ПередЗакрытием(<Отказ>, <ЗавершениеРаботы>, <ТекстПредупреждения>, <СтандартнаяОбработка>)** — Проверка, можно закрыть форму или нельзя. Если отключить стандартную обработку, то форма будет закрыта, независимо от модифицированности<br>  • ЗавершениеРаботы<br>  Тип: Булево.<br>  В параметр передается признак того, что форма закрывается в процессе завершения работы приложения<br>• **ПриЗакрытии(<ЗавершениеРаботы>)** — Выполняется, если форма 100% закрывается. На момент вызова пользователь форму уже не видит<br>  • **ЗавершениеРаботы**<br>  Тип: Булево.<br>  В параметр передается признак того, что форма закрывается в процессе завершения работы приложения: Истина — если в процессе завершения приложения; Ложь — если закрывается только форма. | | | |

### События формы

**? Форма: ОбработкаВыбора(ВыбранноеЗначение, ИсточникВыбора)**
Событие вызывает метод ОповеститьОВыборе(<ЗначениеВыбора>) в подчиненной форме, при открытии которой был установлен Владелец.

**Фокус формы и ТекущийЭлемент**
Активный элемент формы «в фокусе» отслеживается через свойство **ЭтаФорма.ТекущийЭлемент**. Для события изменения текущего элемента формы **обработчик не предусмотрен**.

### Оповещение формы

Всем открытым формам можно передать произвольные данные для любой дальнейшей обработки на их усмотрение. Передача осуществляется вызовом процедуры **Оповестить(ИмяСобытия, Параметр, Источник)**

- **ИмяСобытия** — Строка, содержит идентификатор вида оповещения

Параметр — произвольные данные, передаваемые формам

**Источник** — произвольные данные, передаваемые в качестве источника

Передача оповещения выполняется для тех форм, у которых определен обработчик событий в процедуре **ОбработкаОповещения(ИмяСобытия, Параметр, Источник)**. Процедура обработчика вызывается с теми же параметрами, с которыми была вызвана процедура **Оповестить(…)**.

Кроме открытых форм оповещение может быть обработано модулем приложения или общим глобальным модулем, если оповещение было подключено
процедурой **ПодключитьОбработчикОповещения(<ИмяПроцедуры>)**.

Все описанные процедуры подключения, вызова и обработки выполняются

**&НаКлиенте**:

```
    ПодключитьОбработчикОповещения("СообщитьАдминистратору")
    ...

    Оповестить("ОтправитьСообщение", Новый Структура("Адрес", "abc@gmail.com"),
ЭтаФорма)
    ...
Процедура ОбработкаОповещения(ИмяСобытия, Параметр, Источник)
    Если ИмяСобытия = "ОтправитьСообщение" И Параметр.Адрес = "abc@gmail.com" То
гда
        ОтправитьМейлИзФормы(Источник);
    КонецЕсли;
КонецПроцедуры
...
Процедура СообщитьАдминистратору(ИмяСобытия, Параметр, Источник)
    Если ИмяСобытия = "ОтправитьСообщение" И Параметр.Адрес = "admin@gmail.com"
Тогда
        СообщитьАдминистраторуПоФорме(Источник);
    КонецЕсли;
КонецПроцедуры
```

## Элементы формы

### Поле

Обработчик изменения текста в Поле управляемой формы **ИзменениеТекстаРедактирования(Элемент,Текст,СтандартнаяОбработка)** вызывается для каждого добавления, удаления, вставки или серии таких изменений, после которой последовала пауза больше ~0.2 секунды. Реквизит связанный с полем получает измененное значение только после завершения редактирования, а до завершения актуальное значение изменяемого текста содержит параметр обработчика **Текст** или свойство элемента Поле **.ТекстРедактирования**.

Свойство элемента Поле **.ВыделенныйТекст** содержит фрагмент текста, который выделен, однако перемещение курсора и выделение фрагмента не имеет обработчика, поэтому для его получения необходима

Команда размещенная в Командной панели или в группе вида Командная панель.

**Таблица формы (ТаблицаФормы)**

Таблицы, Поля: ОбработкаВыбора(ВыбранноеЗначение, ИсточникВыбора) — ?

Процедура ПриНачалеРедактирования(Элемент, НоваяСтрока, Копирование) — для вставки начальных значений

Процедура СписокПередНачаломДобавления(Элемент, Отказ, Копирование, Родитель, Группа)

Изменение порядка строк

При использовании команд изменяющих порядок строк в элементе ТаблицаФормы событие не возникает, и существуют только косвенные пути это обнаружить!

**Объекты**

**Виртуальный реквизит .Представление**

Объекты справочников и документов имеют строковую форму представления, которая формируется динамически каждый раз, когда она понадобится, например в модуле во время преобразования типа **Ссылка** к типу Строка или в элементе формы отображающим объект справочника или документа, также строковое представление в запросе доступно через виртуальный реквизит **.Представление**.

Для получения строкового представления платформа реализует алгоритм по умолчанию, который состоит из двух этапов, каждый из которых можно переопределить в модуле менеджера.

Первый этап выполняется в процедуре **ОбработкаПолученияПолейПредставления(Поля,СтандартнаяОбработка)**:

- **Поля** — Массив, необходимо наполнить строковыми именами реквизитов объекта, необходимых для формирования представления
- **СтандартнаяОбработка** — Булево, необходимо установить в **Ложь** при наполнении массива **Поля** значениями
  - если оставить значение **Истина**, то массив Поля будет наполнен стандартной обработкой платформы
    - стандартная обработка для справочников в массиве **Поля** формирует набор значений [«**Наименование**«, «**Ссылка**«]
    - стандартная обработка для документов в массиве **Поля** формирует набор значений [«**Номер**«, «**Дата**«, «**Ссылка**«]

выполнение процедуры **ОбработкаПолученияПолейПредставления(…)** в сеансе кэшируется, т.е. фактически выполняется только один раз

```
Процедура ОбработкаПолученияПолейПредставления(Поля, СтандартнаяОбработка)
    Поля.Добавить("Наименование");
    Поля.Добавить("Версия");
    Поля.Добавить("Ссылка");
    СтандартнаяОбработка = Ложь;
КонецПроцедуры
```

Второй этап выполняется в процедуре **ОбработкаПолученияПредставления(Данные, Представление, СтандартнаяОбработка)**:

- **Данные** — Структура, содержит свойства реквизитов, определенных массивом **Поля** в процедуре **ОбработкаПолученияПолейПредставления(…)**
- **Представление** — Строка, которую необходимо сформировать для представления
  - если **Данные** не содержат нужных реквизитов, их можно получить через их свойства или через ссылку, но это потребует существенное дополнительно время на выполнение
- **СтандартнаяОбработка** — Булево, необходимо установить в **Истина** при наполнении формировании Представления

если оставить значение **Ложь**, то **Представление** будет сформировано базовым алгоритмом платформы

```
Процедура ОбработкаПолученияПредставления(Данные, Представление, СтандартнаяОбра
ботка)
    Представление = Данные.Наименование + Данные.Версия // правильное эффективно
е формирование представления
    + Данные.Версия.ДатаВыпуска     // допустимое, но нежелательное формирование
    + Данные.Ссылка.Комментарий     // неправильное формирование, "Комментарий"
следует добавить в массив Поля
    + Данные.Ссылка.Вид.РеквизитРеквизита;  // и неправильное и очень не желател
ьное формирование
    СтандартнаяОбработка = Ложь;
КонецПроцедуры
```

'''

## Модуль объекта

Обработчики модуля объекта выполняются &НаСервере.
Часть событий открытия и сохранения объекта уже были описаны выше.

ПриКопировании(ОбъектКопирования)

…Аналогично при копировании вызывается обработчик события **ПриКопировании()**, в котором можно дополнить стандартное заполнение данных выполняемое системой при копировании объекта. Этот обработчик вызывается, как при интерактивном копировании, так и при вызове метода **Скопировать()**.

ОбработкаЗаполнения(ДанныеЗаполнения, СтандартнаяОбработка)

При программном открытии формы возможно автоматически заполни ее данными, если в параметр **«Параметры»** включить структуру **«ДанныеЗаполнения»**, и в модуле объекта определить обработку **ОбработкаЗаполнения**(), которая получит структуру **ДанныеЗаполнения** из соответствующего свойства структуры **Параметры**:

```
Процедура ОбработкаЗаполнения(ДанныеЗаполнения, СтандартнаяОбработка)

    Если ДанныеЗаполнения = Неопределено Тогда
        Возврат;

    ИначеЕсли ТипЗнч(ДанныеЗаполнения) = Тип("Структура") Тогда
        ЗаполнитьЗначенияСвойств(ЭтотОбъект, ДанныеЗаполнения);

    ИначеЕсли Метаданные().ВводитсяНаОсновании.Содержит(ДанныеЗаполнения.Метадан
ные()) Тогда
        ЗаполнитьПоДокументуОснованию(ДанныеЗаполнения);

    КонецЕсли;

КонецПроцедуры
```

## Оповещение в асинхронной модели

Классические диалоговые окна, приводящие к блокировке потока приложения, являются анахронизмом, который достаточно просто можно заменить механизмом оповещения.

## Оповещение форм и модулей

Механизм оповещения позволяет первичной форме открыть вторичную форму и получить оповещение при ее закрытии:

- первичная форма может передать вторичной любые данные используя **Параметр** открытия
- оповещение первичной формы будет сделано путем вызова процедуры, заданной в **ОписаниеОповещения**
  - имя процедуры определяет параметр описания **ИмяПроцедуры**
  - оповещение получит первичная форма, если использует **ЭтотОбъект** или другая форма или модуль указанная в параметре **Модуль**
  - процедура оповещения может может получить произвольные данные от первичной формы через параметр **ДополнительныеПараметры**

- вторичная форма получает данные открытия через структуру **Параметры**
- результирующие данные вторичная форма передает первичной через параметр процедуры **Закрыть()**
- первичная форма должна иметь экспортную процедуру с заданным именем и двумя параметрами
- первый параметр оповещения содержит данные помещенные в вызов **Закрыть()**

второй параметр содержит данные оповещения из **ДополнительныеПараметры**

```
&НаКлиенте //команда первичной формы открывает вторичную форму, передает данные
и ключ
Процедура КомандаПервичной(Команда)
    ОткрытьФорму("ОбщаяФорма.ФормаВторичная",
        Новый Структура("ДанныеПервичной", ДанныеПервичной),,,,,
        Новый ОписаниеОповещения("ОповещениеПервичной", ЭтотОбъект, ДопДанныеПер
вичной),
        РежимОткрытияОкнаФормы.БлокироватьОкноВладельца);
    ...

&НаСервере // вторичная форма получает первичные данные
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)
    ПараметрыВторичной = Параметры.ДанныеПервичной;
    ...

&НаКлиенте // вторичная форма
Процедура КомандаВторичной(Команда)
    Закрыть(ДанныеВторичной);
    ...

&НаКлиенте
Процедура ОповещениеПервичной(ДанныеВторичной, ДопДанныеПервичной) Экспорт
    ...
```

## Оповещение функций Показать…

Семейство объединяет функции для асинхронного выбора и ввода данных пользователя при работе с формой, в их числе: ПоказатьВопрос, ПоказатьВводЗначения, ПоказатьВводЧисла, ПоказатьВводСтроки, ПоказатьВводДаты, ПоказатьВыборИзМеню, ПоказатьВыборИзСписка, ПоказатьВыборЭлемента, ПоказатьЗначение, ПоказатьИнформациюОбОшибке, ПоказатьОповещениеПользователя, ПоказатьОтметкуЭлементов, ПоказатьПредупреждение.

Общая схема их применения показана на примере, в котором Вопрос оповещает форму об ответе и передает через себя Таблицу данных:

```
&НаКлиенте
Процедура КомандаОчистить(Команда)
    ПоказатьВопрос(
        Новый ОписаниеОповещения("ОтветОчистить", ЭтотОбъект, ТаблицаДанных),
        "Очистить таблицу?", РежимДиалогаВопрос.ОКОтмена,, КодВозвратаДиалога.От
мена);
    ...

&НаКлиенте
Процедура ОтветОчистить(Ответ, ТаблицаДанных) Экспорт
    Если Ответ = КодВозвратаДиалога.ОК Тогда
        ТаблицаДанных.Очистить();
        ...
```

### Обработчик ожидания

Обработка ожидания специфична тем, что не связана с какими-либо событиями НаКлиенте или НаСервере. События обработчика ожидания происходят по таймеру с заданным интервалом с момента его подключения процедурой **ПодключитьОбработчикОжидания(<ИмяПроцедуры>, <Интервал>, <Однократно>)** и до момента его отключения процедурой **ОтключитьОбработчикОжидания(<ИмяПроцедуры>)**. Пример:

```
&НаКлиенте
...
ПодключитьОбработчикОжидания("МойОбработчик", 30, Ложь);
...
ОтключитьОбработчикОжидания("МойОбработчик");
...

&НаКлиенте
Процедура МойОбработчик()
    ...
КонецПроцедуры
```

- параметр **<ИмяПроцедуры>** задает процедуру обработчика, для которого возможны два контекста (суть и синтаксис в обоих контекстах не отличаются)
  - **в глобальном контексте** обработчиком может быть подключена любая **процедура глобального общего модуля** без параметров компилируемая НаКлиенте
  - **в модуле формы** обработчиком может быть подключена любая процедура модуля без параметров
    - обработчик ожидания в каждой форме вызывается независимо от других обработчиков в других формах
    - при закрытии формы вызовы ее обработчиков прекратятся
  - процедура обработчика ожидания выполняется НаКлиенте, но имеет возможность выполнять любые вызовы, в т.ч. **НаСервере**
- параметр **<Интервал>** устанавливает задержку каждого ожидания в секундах, которая может быть произвольной, но точность ее соблюдения составляет 0.1 секунды

- интервал **менее 1.0 секунды** предполагает **однократный** вызов обработчика

- параметр **<Однократно>** равный **Истина** приведет к однократному вызову обработчика и не требует Отключения

  Вы познакомились с обрабочиками. Если у вас остались сомнения, на каком клиенте данный обработчик должен выполняться на просторах интернета есть все или почти все. Посмотрите видео лекцию для полного понимания  данной темы.

  Канал **техподдержка**

**Что такое Контекст в 1С Программировании и как в нем ориентироваться?! Урок 16**
**https://www.youtube.com/watch?v=O1nD8STnB5s**
**Желаю успехов!**