Отказ от использования модальных окон в платформе “1С:Предприятие 8.3”

При разработке конфигурации на платформе *1С:Предприятие 8* периодически возникает потребность приостановить работу программы до того момента, когда пользователь примет какое-либо решение или выполнит какие-либо действия.

Например, при нажатии на кнопку заполнения табличной части у пользователя следует спросить, нужно ли очистить табличную часть, чтобы не произошло потери ранее введенных данных.

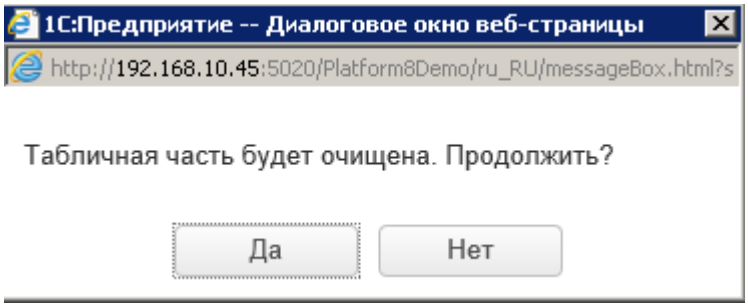
Такое поведение может обеспечить, например, следующий код:

```
&НаКлиенте
Процедура ЗаполнитьТовары(Команда)

    Ответ = Вопрос("Табличная часть будет очищена. Продолжить?",
РежимДиалогаВопрос.ДаНет);
    Если Ответ = КодВозвратаДиалога.Да Тогда
    //алгоритм заполнения
    КонецЕсли;

КонецПроцедуры
```

В результате работы этого фрагмента кода произойдет приостановка выполнения программного кода, на экране отображается вопрос, интерфейс приложения кроме диалога с вопросом становится недоступным, система ожидает принятия решения пользователем, выполнение кода продолжится только после ответа на вопрос.

Также к приостановке выполнения кода и блокировке интерфейса приводит открытие модальных окон при помощи вызова метода *ОткрытьМодально()*.

При работе с конфигурацией в режиме веб-клиента через браузер в этом случае будет открыто новое окно – всплывающее окно, которое заблокирует не только текущую вкладку, но и весь интерфейс браузера, включая остальные открытые окна и вкладки.



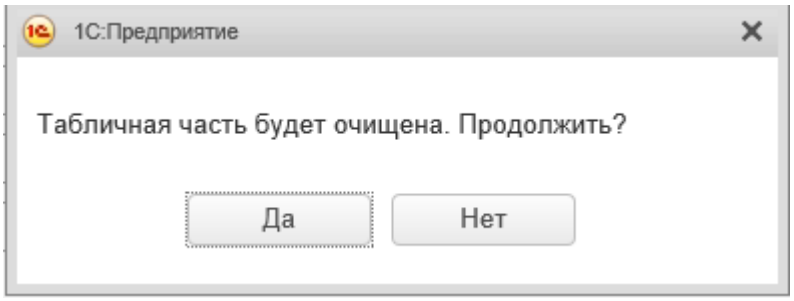
Всплывающие окна в Интернете зачастую используются для злоумышленного распространения нежелательной рекламы, поэтому браузеры содержат функции блокировки всплывающих окон.

В таком случае для работы с конфигурациями *1С:Предприятие 8* через браузер необходимо запретить блокирование всплывающих окон.

Проблемы также возникают при работе на мобильных устройствах. Так, например, модальные окна не поддерживаются на *iPad*.

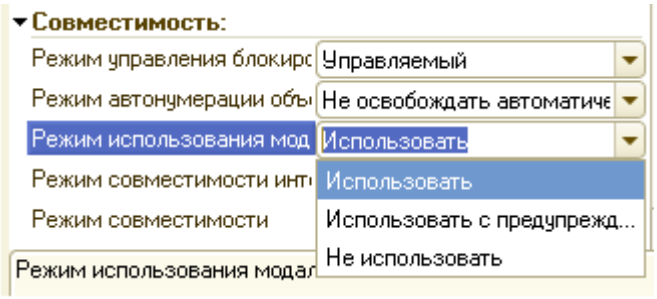
Для решения указанных проблем следует использовать блокирующие окна вместо модальных. Для пользователя визуально все выглядит так же: окно блокирует интерфейс веб-клиента.

Однако блокирующее окно как бы “рисуется” поверх главного окна, и блокируется только текущая вкладка браузера, в которой открыта конфигурация, позволяя переключаться на другие вкладки, поскольку модальные окна браузера при этом не используются.



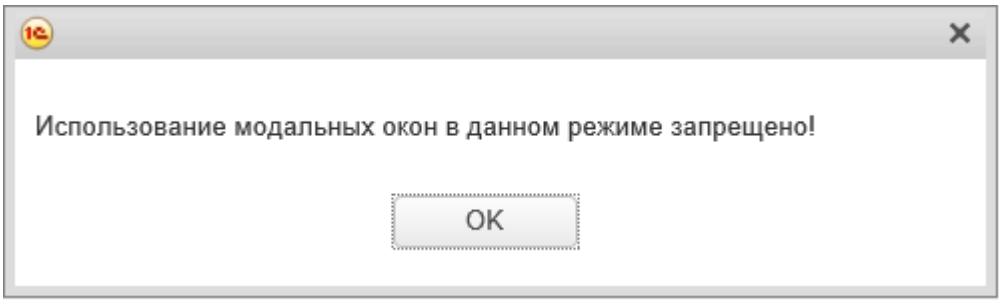
Таким образом, всплывающие окна в браузере не открываются и обеспечивается работа через веб-клиент на мобильных устройствах.

У корневого элемента конфигурации существует свойство *“Режим использования модальности”*, которое определяет, можно ли в конфигурации открывать модальные окна.



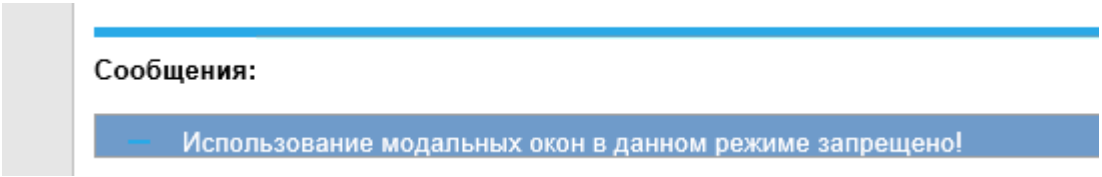
Если выбран вариант *“Использовать”*, то модальные окна можно открывать. Если выбран вариант *“Не использовать”*, то модальные окна недопустимы.

При попытке вызвать метод, открывающий модальное окно, система выводит сообщение об ошибке:



При таком значении свойства *“Режим использования модальности”* допустимы только блокирующие окна.

Если выбран вариант *“Использовать с предупреждениями”*, то при открытии модальных окон в окно сообщений выводится текст:



Такой вариант работы может использоваться как промежуточный при переработке конфигурации с целью отказа от использования модальных окон.

Основное отличие блокирующих окон от модальных заключается в том, что открытие блокирующего окна не производит приостановки выполнения кода.

Поэтому разработчикам придется переписать программный код, использующий модальные окна, с учетом этой особенности.

Код нужно разделить на две части:

- открытие блокирующего окна;
- обработка выбора пользователя.

Фрагмент кода, приведенный в начале статьи, нужно переписать следующим образом:

```
&НаКлиенте
Процедура ЗаполнитьТовары(Команда)

    Оповещение = Новый ОписаниеОповещения("ЗаполнитьТоварыВопросЗавершение",
ЭтотОбъект);
    ТекстВопроса = "Табличная часть будет очищена. Продолжить?";
    ПоказатьВопрос(Оповещение, ТекстВопроса, РежимДиалогаВопрос.ДаНет);

КонецПроцедуры

&НаКлиенте
Процедура ЗаполнитьТоварыВопросЗавершение(Результат, ДополнительныеПараметры) Экспорт

    Если Результат = КодВозвратаДиалога.Да Тогда
            //алгоритм заполнения
    КонецЕсли;

КонецПроцедуры
```

После выполнения процедуры *ПоказатьВопрос()* система не останавливается, ожидая ответ пользователя, исполнение кода продолжается. Пользователь сможет сделать выбор только после завершения работы всей процедуры.

При этом будет вызвана экспортная процедура *ЗаполнитьТоварыВопросЗавершение*(). Ее название мы передали в конструктор объекта *ОписаниеОповещения*.

Процедура, которая будет вызвана после осуществления выбора, может быть расположена в модуле формы, модуле команды, общем не глобальном модуле.

В рассмотренном примере вызываемая процедура расположена в модуле управляемой формы, поэтому мы передали в параметр *ЭтотОбъект*.

Рассмотрим вызов процедуры, расположенной в общем модуле. Для этого добавим новый общий модуль *ОбработкаОповещений*, установим для него флаг *“Клиент (управляемое приложение)”*, а признак *“Глобальный”* не устанавливаем.

Расположим в этом модуле процедуру *ЗаполнитьТоварыВопросЗавершение()*.

Тогда обработчик команды заполнения будет выглядеть так:

```
&НаКлиенте
Процедура ЗаполнитьТовары(Команда)

	Оповещение = Новый ОписаниеОповещения("ЗаполнитьТоварыВопросЗавершение",
	ОбработкаОповещений);
	ТекстВопроса = "Табличная часть будет очищена. Продолжить?";
	ПоказатьВопрос(Оповещение, ТекстВопроса, РежимДиалогаВопрос.ДаНет);

КонецПроцедуры
```

После вызова любого метода, открывающего блокирующее окно, процедура должна завершаться, а выполняемый далее код следует располагать в процедуре, которая будет вызвана после закрытия окна.

Для передачи контекста (вспомогательных данных, неких параметров, значений переменных) из процедуры, открывающей модальное окно, в процедуру, вызывающуюся при его закрытии, предусмотрен третий необязательный параметр конструктора объекта *ОписаниеОповещения* - *ДополнительныеПараметры*.

Этот объект (любого типа) будет передан в процедуру, описанную в *ОписаниеОповещения*, последним параметром.

На примере рассмотренного выше участка кода это можно сделать так:

```
&НаКлиенте
Процедура ЗаполнитьТовары(Команда)

	Параметр1 = 0;
	Параметр2 = 0;
	СписокПараметров = Новый Структура("Параметр1, Параметр2", Параметр1,
Параметр2);

	Оповещение = Новый ОписаниеОповещения("ЗаполнитьТоварыВопросЗавершение",
ЭтотОбъект, СписокПараметров);
	ПоказатьВопрос(Оповещение, "Табличная часть будет очищена. Продолжить?",
РежимДиалогаВопрос.ДаНет);

КонецПроцедуры

&НаКлиенте
Процедура ЗаполнитьТоварыВопросЗавершение(Результат, ДополнительныеПараметры) Экспорт

	Если Результат = КодВозвратаДиалога.Да Тогда
	//анализируем ДополнительныеПараметры.Параметр1
	//анализируем ДополнительныеПараметры.Параметр2
	КонецЕсли;
КонецПроцедуры
```

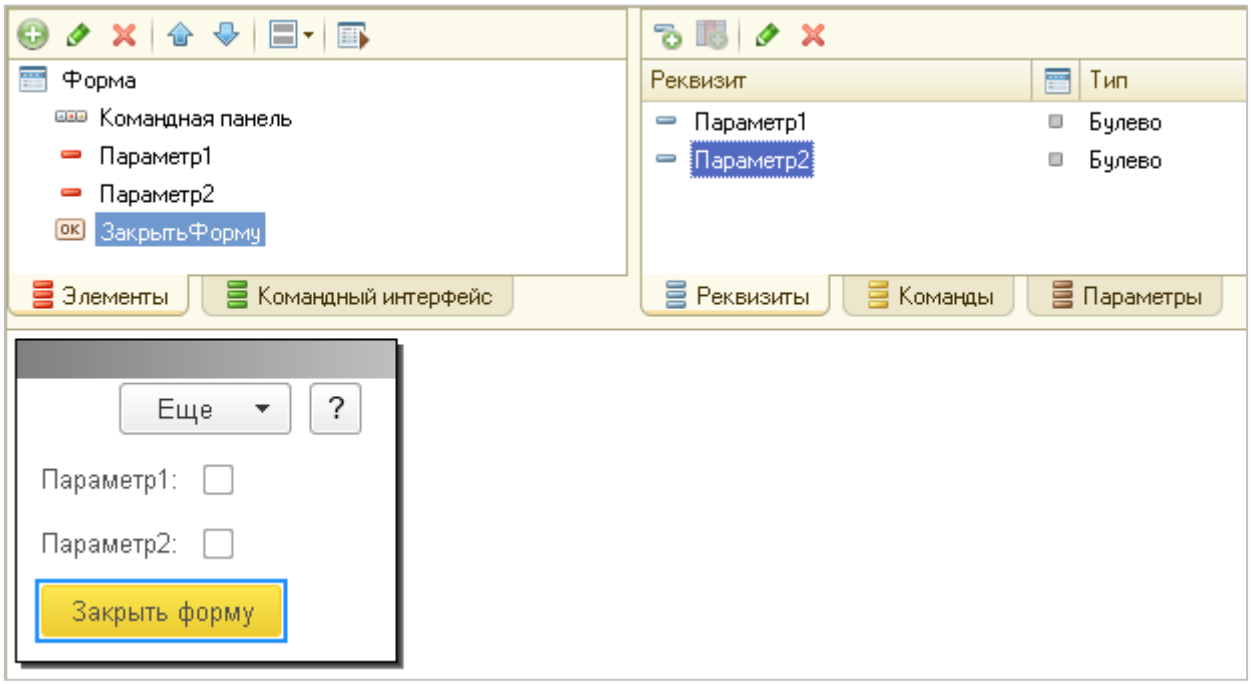
Если нужно передать только одно значение, то структуру можно не использовать, а присвоить это значение параметру *ДополнительныеПараметры* конструктора объекта *ОписаниеОповещения*. Рассмотрим несколько примеров работы с блокирующими окнами.

Задача 1. Открытие другой формы

Из формы документа по нажатию на кнопку "Открыть параметры" нужно открыть форму, на которой расположены два флажка Параметр1 и Параметр2, которые должен установить пользователь.

После закрытия формы вывести в строку сообщений значения параметров.

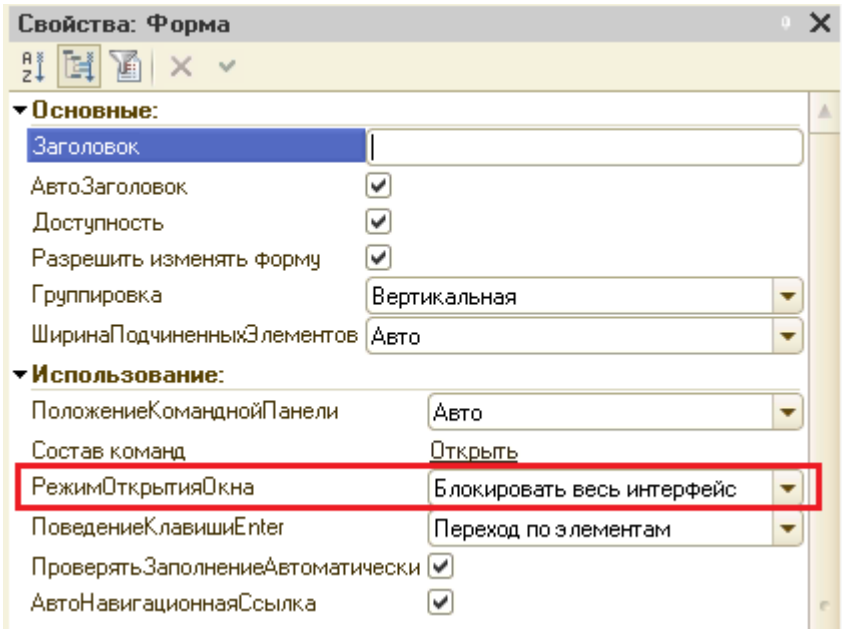
Создаем общую форму *"ФормаПараметров"*, на которой размещаем реквизиты *Параметр1* и *Параметр2*, а также команду *ЗакрытьФорму*:



Обработчик команды выглядит следующим образом:

```
&НаКлиенте
Процедура ЗакрытьФорму(Команда)
    СписокПараметров = Новый Структура("Параметр1, Параметр2", Параметр1, Параметр2);
    Закрыть(СписокПараметров);

КонецПроцедуры
```

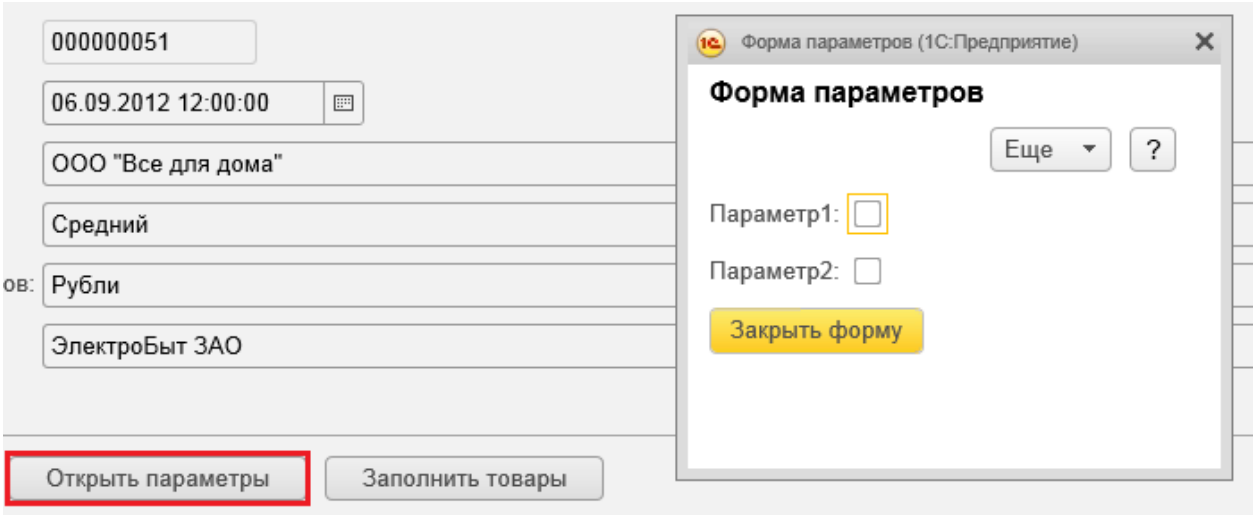
Для формы свойство *РежимОткрытияОкна* устанавливаем в "Блокировать весь интерфейс":

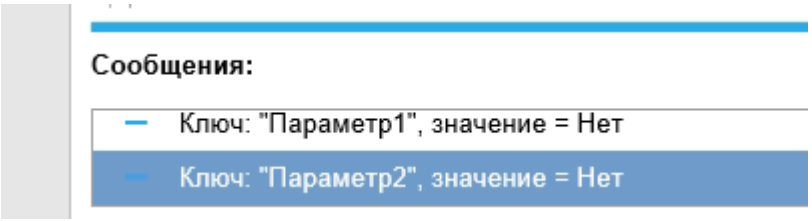


На форме документа располагаем команду *ОткрытьПараметры*, обработчик которой описываем следующим образом:

```
&НаКлиенте
Процедура ОткрытьПараметры(Команда)

    Оповещение = Новый ОписаниеОповещения("ОткрытьПараметрыЗавершение", ЭтотОбъект);
    ОткрытьФорму("ОбщаяФорма.ФормаПараметров", , , , , , Оповещение);

КонецПроцедуры

&НаКлиенте
Процедура ОткрытьПараметрыЗавершение(Результат, ДополнительныеПараметры) Экспорт

    Если ТипЗнч(Результат) = Тип("Структура") Тогда

     Для каждого КлючЗначение Из Результат Цикл

        Сообщение = Новый СообщениеПользователю;
        Сообщение.Текст = "Ключ: """ + КлючЗначение.Ключ + """, значение = " +
КлючЗначение.Значение;
         Сообщение.Сообщить();

     КонецЦикла;

    КонецЕсли;

КонецПроцедуры
```

В пользовательском режиме, запуская конфигурацию под веб-клиентом, получаем такие результаты работы:





Режим открытия окна можно также указывать в последнем параметре процедуры *ОткрытьФорму*.

```
&НаКлиенте
Процедура ОткрытьПараметры(Команда)

    Оповещение = Новый ОписаниеОповещения("ОткрытьПараметрыЗавершение", ЭтотОбъект);
    ОткрытьФорму("ОбщаяФорма.ФормаПараметров", , , , , , Оповещение,
РежимОткрытияОкнаФормы.БлокироватьВесьИнтерфейс);

КонецПроцедуры
```

Задача 2. Вопрос при закрытии формы

При закрытии окна обработки задавать пользователю вопрос, действительно ли он хочет закрыть окно.

Эту задачу можно решить при помощи следующего кода, расположенного в модуле формы обработки:

```
&НаКлиенте
Перем НужноЗакрыватьФорму;

&НаКлиенте
Процедура ПередЗакрытием(Отказ, СтандартнаяОбработка)
```

```
    Если НЕ НужноЗакрыватьФорму = Истина Тогда
    Отказ = Истина;
    Оповещение = Новый ОписаниеОповещения("ПередЗакрытиемЗавершение", ЭтотОбъект);
    ПоказатьВопрос(Оповещение, "Вы действительно хотите закрыть окно?",
РежимДиалогаВопрос.ДаНет);
    КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ПередЗакрытиемЗавершение(Результат, ДополнительныеПараметры) Экспорт
    Если Результат = КодВозвратаДиалога.Да Тогда
     НужноЗакрыватьФорму = Истина;
     Закрыть();
    Иначе
     НужноЗакрыватьФорму = Неопределено;
    КонецЕсли;
КонецПроцедуры
```

В процедуре *ПередЗакрытием* формы пользователю задается вопрос, флаг *Отказ* выставляется в *Истина*, закрытие формы отменяется. После утвердительного ответа на вопрос переменная *НужноЗакрыватьФорму* устанавливается в *Истина*, форма закрывается повторно.

Задача 3. Ввод числового значения

При нажатии на кнопку на форме обработки открывать стандартный диалог ввода числа.

Для этого необходимо воспользоваться методом *ПоказатьВводЧисла()* вместо *ВвестиЧисло()*, который открывает блокирующее окно вместо модального.

```
&НаКлиенте
Процедура ВводЧисла(Команда)
    Оповещение = Новый ОписаниеОповещения("ВводЧислаЗавершение", ЭтотОбъект);
    ПоказатьВводЧисла(Оповещение, 0, "Введите количество", 15, 3);
КонецПроцедуры

&НаКлиенте
Процедура ВводЧислаЗавершение(Результат, ДополнительныеПараметры) Экспорт
    Если НЕ Результат = Неопределено Тогда
     Сообщение = Новый СообщениеПользователю;
     Сообщение.Текст = "Вы ввели количество " + Результат;
     Сообщение.Сообщить();
    КонецЕсли;
КонецПроцедуры
```

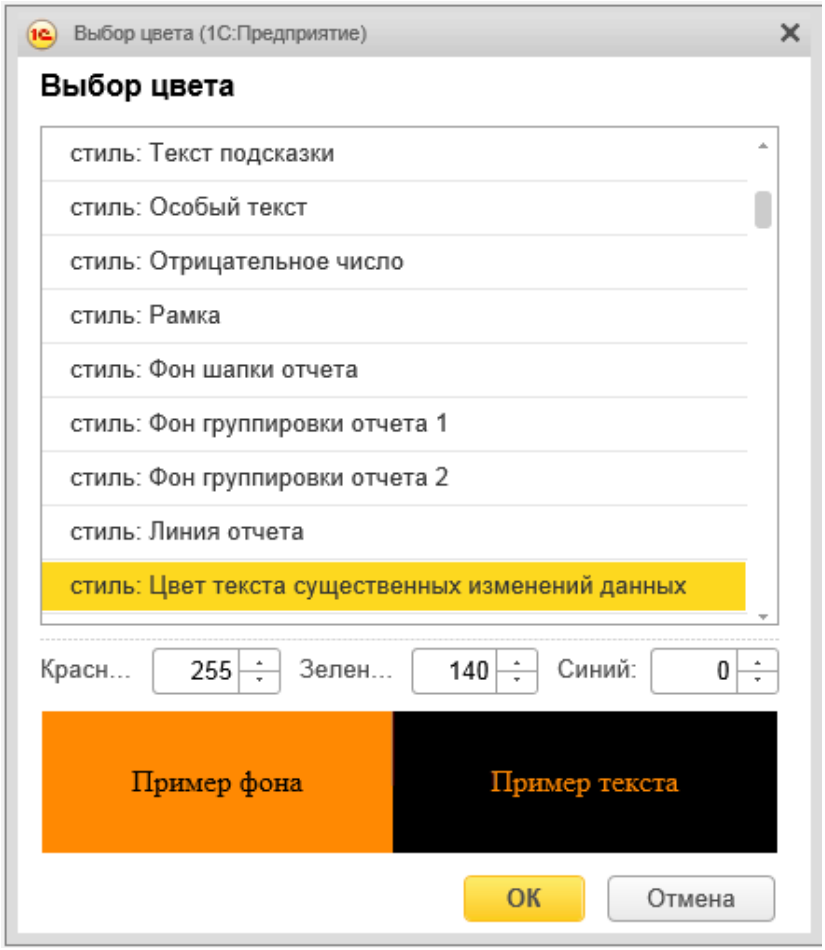
После закрытия окна ввода числа будет вызвана процедура, в первый параметр которой будет передано введенное число или значение *Неопределено*, если пользователь отказался от ввода.

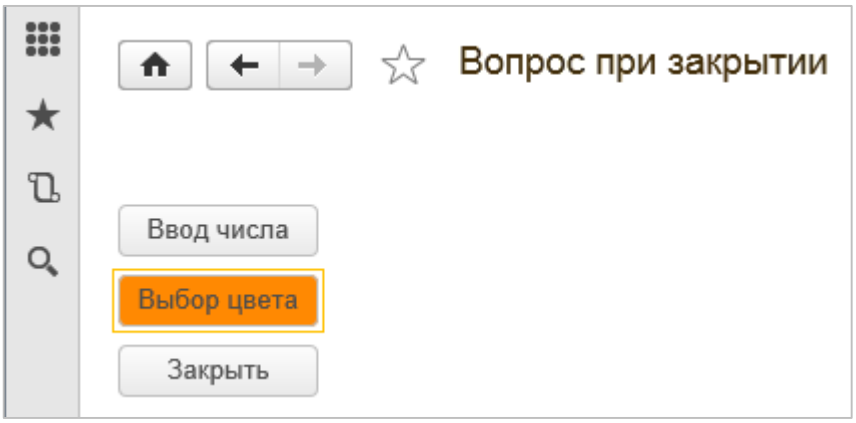
Задача 4. Выбор цвета

При нажатии на кнопку на форме обработки при помощи стандартного диалога выбора цвета пользователь указывает необходимый цвет. Этот цвет установить для фона нажимаемой кнопки.

Добавим на форму команду *ВыборЦвета* со следующим обработчиком:

```
&НаКлиенте
Процедура ВыборЦвета(Команда)
    ДиалогВыбораЦвета = Новый ДиалогВыбораЦвета;
    Оповещение = Новый ОписаниеОповещения("ВыборЦветаЗавершение", ЭтотОбъект);
    ДиалогВыбораЦвета.Показать(Оповещение);
КонецПроцедуры

&НаКлиенте
Процедура ВыборЦветаЗавершение(Результат, ДополнительныеПараметры) Экспорт
    Если НЕ Результат = Неопределено Тогда
     Элементы.ВыборЦвета.ЦветФона = Результат;
    КонецЕсли;
КонецПроцедуры
```





Для объектов *ДиалогВыбораЦвета* (а также *ДиалогРедактированияСтандартногоПериода*, *КонструкторФорматнойСтроки*, *ДиалогРасписанияРегламентногоЗадания*, *ДиалогВыбораШрифта*) метод *Показать()* открывает блокирующее окно.

После закрытия окна будет вызвана процедура, в первый параметр которой будет передано выбранное значение (цвет, шрифт и т.д.) или значение *Неопределено*, если пользователь отказался от выбора.

Следует обратить внимание, что объект *ДиалогВыбораФайла* не имеет метода *Показать()* в отличие от диалогов выбора цвета или шрифта, поскольку реализация этих диалогов существенно разная.

Для использования диалога выбора файла на веб-клиенте необходимо предварительно подключить расширение работы с файлами.

Диалоги, реализуемые через расширение работы с файлами, не создают таких проблем в работе, как модальные окна браузеров, поэтому не было реализовано открытие блокирующих окон для объекта *ДиалогВыбораФайла*.

Ханевич Василий

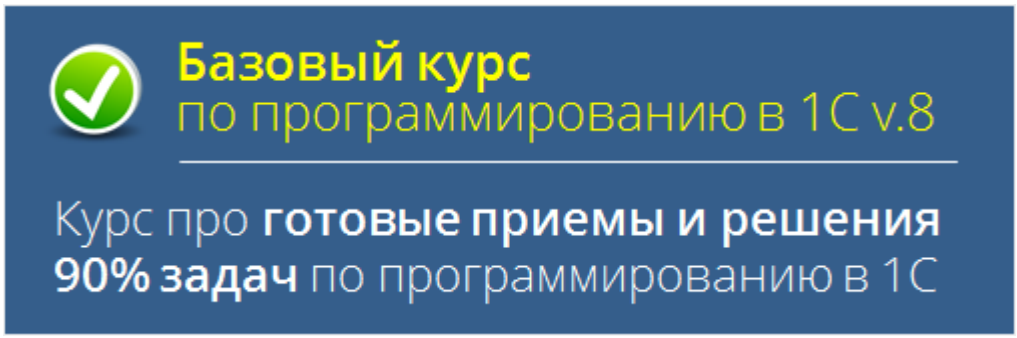
г. Калининград

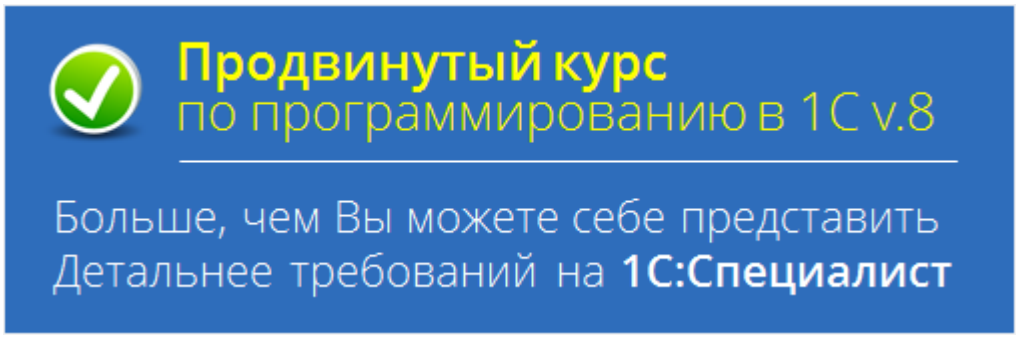
Дополнительные материалы

Все статьи проекта **Курсы-по-1С.рф**: **http://курсы-по-1с.рф/blog/articles/**

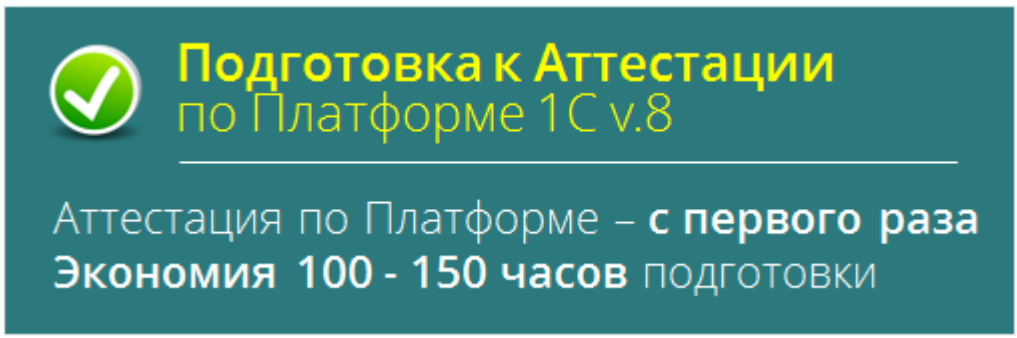
Курсы по программированию в 1С v.8

Базовый и Продвинутый курсы по Программированию на Платформе 1С 8
http://www.Spec8.ru/





«Курс по подготовке к Аттестации по Платформе 1С 8.2 / 8.3»
http://курсы-по-1с.рф/dev-attestation/



Бесплатные материалы проекта:
http://курсы-по-1с.рф/free/

Электронная почта: support@Kursy-po-1C.ru
Телефоны: **8-800 1000-613**, +7(495) 7777-096