

Интерактивное взаимодействие и оповещения

Вывод информации пользователю

```
Предупреждение("Текст", Таймаут_число, "Заголовок окна сообщения"); //окно сообщения  
Сообщить("1!", СтатусСообщения.Важное); //в блоке сообщений
```

Получение информации от пользователя во всплывающем окне

```
ВвестиЧисло();  
ВвестиЗначение();  
ВвестиСтроку();  
ВвестиДату();
```

Вариации модальных окон (предыдущие примеры) использовать нежелательно, скорее всего от данной технологии будут отказываться. Лучше использование асинхронного взаимодействия. В частности, можно использовать процедуру “ПоказатьВопрос()” в сочетании с механизмом описания оповещения.

```
&НаКлиенте  
Процедура СоздатьДоговоры(Команда)  
□Если ЭтотОбъект.Модифицированность Тогда  
□□ОписаниеОповещения = Новый ОписаниеОповещения("СоздатьДоговорыЗавершение", ЭтотОбъект);  
□□ПоказатьВопрос(ОписаниеОповещения, "Документ должен быть записан. Записать и продолжить?",  
РежимДиалогаВопрос.ДаНет);  
□Иначе  
□□Если спрашивать не требуется вызываем процедуры завершения  
□□СоздатьДоговорыЗавершение(КодВозвратаДиалога.Нет);□  
□КонецЕсли;  
КонецПроцедуры  
  
&НаКлиенте
```

```
Процедура СоздатьДоговорыЗавершение(Результат, Параметры = Неопределено) Экспорт
□Если Результат = КодВозвратаДиалога.Да Тогда
□□//Действия выполняемые при положительном ответе пользователя
□КонецЕсли; □
□□//Действия выполняемые независимо от ответа пользователя
КонецПроцедуры
```

Обновление данных на форме в пределах своей сессии

Оповещения

[Статья про оповещения пользователю](#)

Под термином "Оповещение" в одном контексте оповещения пользователю (ссылка выше), в другом - способ взаимодействия между формами в пределах одной сессии (это важно!). Для второго варианта в форме пишется обработчик оповещений, при генерации в форме-источнике запускается обработчик и в обработчике можно фильтровать события по источнику и названию. Типа шина межоконного клиентского взаимодействия в пределах одной сессии.

```
//В первой форме оповещаются все открытые формы
Оповестить("ИмяСобытия", Данные, "Форма1");

//В другой форме следует назначить обработчик события оповещения.
Процедура ОбработкаОповещения(Событие, Данные, Источник)
    Если Событие = "ИмяСобытия" И Источник = "Форма1" Тогда
        КонецЕсли;
    КонецПроцедуры
```

И - нихера не заработало! А все потому, что процедуру ОбработкаОповещения нужно создавать не лапками, а через ПКМ на форме - События - ОбработкаОповещения либо указав созданную лапками процедуру в событии ОбработкаОповещения. И реально работает только если форма открыта.

Обновление данных на форме у всех пользователей

Здесь кроется одна из фундаментальных проблем 1С, понять и простить. При открытии формы редактирования (например элемента справочника, ... будем называть его объектом), до версии 8.3.14 объект блокировался на изменение у других пользователей. Т.е., если я открыл в справочнике Номенклатура объект Ручка красная, то если кто-то попытается его открыть у себя - получит сообщение о блокировке на запись и все. И пока я просто держу открытой эту вкладку- она заблокирована у всех остальных. После 8.3.14 стало чуть лучше, блокировка для данных открытой формы снимается через 1 минуту после сохранения. Работает на толстом клиенте в файловом режиме. Однако в web клиенте в файловом

режиме блокировка устанавливается после первого нажатия Сохранить, затем не снимается до закрытия формы. Подробнее о [механизме блокировок](#)

Динамические списки: Элементы.ИмяДинамическогоСписка.Обновить()

Реквизиты формы: Методы ЭтаФорма.Прочитать() и ЭтаФорма.ОбновитьОтображениеДанных(). Метод Прочитать обновляет объект управляемой формы. Метод ОбновитьОтображениеДанных принудительно обновляет содержание элементов управления, особенно полезно, когда изменения значений реквизитов формы происходят вне самой формы. Это гарантирует, что каждый элемент в форме будет соответствовать последним изменениям данных в системе.

Динамическое обновление открытой формы. Методы в предыдущем разделе должны быть чем-то вызваны. Периодические задания исполняются только на сервере, для клиента не подходят. [Шикарная статья про обработчики](#) Понять и простить. С ностальгией вспомни школьные годы, метод обновления периодическими опросами сервера и вперед! Базовый код:

```
&НаКлиенте
Процедура ОболочкаОбработчика()
□Элементы.СписокГородов.Обновить();
КонецПроцедуры

&НаКлиенте
Процедура ПодключитьОжидание(Команда)
□ПодключитьОбработчикОжидания("ОболочкаОбработчика", 10, Ложь);
КонецПроцедуры

&НаКлиенте
Процедура ОтключитьОжидание(Команда)
□ОтключитьОбработчикОжидания("ОболочкаОбработчика");
КонецПроцедуры
```

Это работает и в web клиенте. Не нашел способа получить список подключенных обработчиков, поэтому необходимы глазками-контролируемые правила создания и отключения обработчиков.

Важно: Если действие на сервере затратило меньше времени, чем таймаут, то обработчик строго по таймауту вызовет нужное, будто серверной приостановки и не было; а если больше, то "собьётся" и вызовется сразу по возвращении с сервера.

При высокой нагрузке на базу стоит усложнить алгоритм см. [Динамические списки](#) раздел Обновление данных.

Пример автоматического обновления данных в таблице значений.

Есть справочник Города, только Наименование. Есть обработка, где в таблице будем выводить и обновлять значения из справочника. Практически это может пригодиться только если форма объемная, нужно поддерживать актуальность одной таблицы из нескольких, а полное обновление формы замедлит работу.

У элемента Таблица нужно добавить колонки реквизитов, в этом случае - одну колонку Наименование. При добавлении на форму, будет предложено добавить колонки.

Вроде не может использоваться на веб клиенте, но у меня поехало.

&НаКлиенте

Процедура ОболочкаОбработчика()

□//Элементы.СписокГородов.Обновить();

□ОбновитьТаблицу();

КонецПроцедуры

&НаКлиенте

Процедура ПодключитьОжидание(Команда)

□ПодключитьОбработчикОжидания("ОболочкаОбработчика", 10, Ложь);

КонецПроцедуры

&НаКлиенте

Процедура ОтключитьОжидание(Команда)

□ОтключитьОбработчикОжидания("ОболочкаОбработчика");

КонецПроцедуры

&НаСервере

Процедура ОбновитьТаблицу();

□Запрос = Новый Запрос;

□Запрос.Текст =

□"ВЫБРАТЬ

□□Города.Наименование КАК Наименование

□□ИЗ

□□Справочник.Города КАК Города";

□РезультатЗапроса = Запрос.Выполнить();

□//СпособОбхода = ОбходРезультатаЗапроса.Прямой;

□СписокГородовТаблица.Загрузить(РезультатЗапроса.Выгрузить())

КонецПроцедуры

&НаСервере

Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

ОбновитьТаблицу();

КонецПроцедуры

Метод Загрузить(РезультатЗапроса.Выгрузить) работает если наименования колонок совпадают с наименованиями столбцов в запросе.

Заметил, что при обновлении динамического списка на указателе мыши возникает кружок ожидания, однако фокус в списке после обновления не смещается. У таблицы кружок не возникает, однако фокус смещается на первый элемент.

На потом, не завершено: Появилась идея в обновления таблицы только в случае изменения хэша. Нужно бы проверить скорость работы, но пока не до этого. Однако сам факт вычисления хэша затратен.

На потом, не завершено: Предположительный алгоритм обновления с динамическим изменением периода опроса.

Проблема: в случае большого количества клиентов со включенными обработчиками возможна нагрузка на сервер.

Задача: Необходим алгоритм, управляющий периодами обновления данных на клиентах с приоритетами по объектам. Также при необходимости (например, наличие ресурсоемких задач на сервере), должен быть способ общего увеличения периода. На сервере могут быть другие базы, иные серверные скрипты, поэтому у нас нет полного доступа ко всем процессам, только локальная оценка производительности. Все клиенты равнозначны (хотя для общего решения лучше делить по пользователям, но потом).

Решение:

На сервере:

Регистр сведений, измерения - НазваниеПроцедуры, Ресурсы - ВремяПоследнегоИзменения, ЗадержкаОбновления. Служит для серверного изменения при обновлении данных. Способ реализации обновления этого регистра - отдельный вопрос.

Зарезервированное название процедуры - ПериодОбновленияРегистра для управления обновлением регистра на клиенте. При нагрузке увеличить задержку обновления и общий период обновления.

На клиенте:

ПараметрСеанса Строка, сериализованный аналог полей регистра сведений об обновлениях, только индивидуально для каждого сеанса. Часть процедур, которые нужны конкретному сеансу. На формы добавить события ПриОткрытии/Закрытии, для поддержания актуальности о подписанных процедурах. Процедуры сериализации / десериализации данных и обновления регистра. При пустом параметре, обновления регистра прекращаются. Если ВремяПоследнегоИзменения в регистре больше чем в параметре сеанса, то через

ЗадержкуОбновления произвести обновление. Т е у каждой формы практически одинаковый алгоритм проверки / обновления.

Вопросы к постановке задачи и алгоритму:

1. Как определить, что количество клиентов стало "большим"
2. А может быть это не нужно?

Revision #19

Created 7 January 2025 17:22:51 by Admin

Updated 11 February 2025 17:07:30 by Admin