JavaScript-библиотека. Возможности и примеры использования.

Конференция 1С-Битрикс – 26 января 2012

Антон Герасимюк

1С·БИТРИКС

О чем доклад?

- Архитектура библиотеки
- Основные функции и методы
- Обзор расширений библиотеки
  - Ajax
  - Анимация
  - Работа с LocalStorage
  - Оконная библиотека
  - Форматирование даты
- Демонстрация примеров

1С·БИТРИКС

Предыстория создания

- 5 лет назад в Битриксе JS-кода было очень мало (админка, визуальный редактор, календарь)
- Развитие Web потребовало более продвинутых интерфейсов.
- Количество Javascript’а стало постоянно увеличиваться. Код был разрозненным и часто дублировался.
- JS-библиотека объединила все старые наработки и упростила разработку новых модулей.
- Сейчас в новых модулях Битрикса Javascript-кода больше чем PHP-кода

1С·БИТРИКС

Зачем свой велосипед?

- Библиотека учитывает особенности Битрикса
- Есть совместимость со старым кодом
- Никаких проблем с поддержкой и расширением функциональности
- JS-библиотека не накладывает никаких ограничений - используйте совместно свой любимый фреймворк.

1С·БИТРИКС

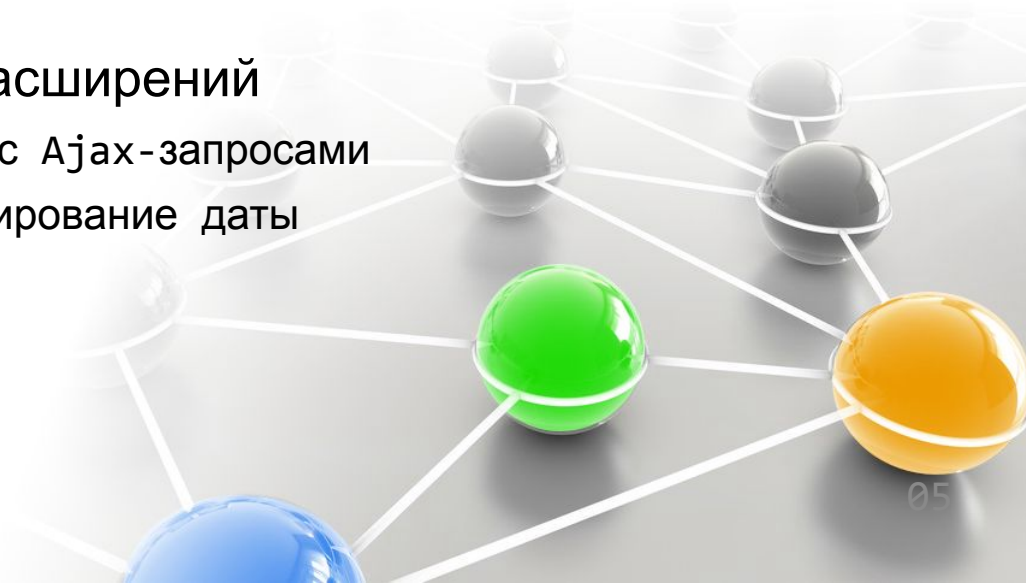
Архитектура

- Все файлы библиотеки находятся в папке `/bitrix/js/main/core/`

- `core.js` – ядро библиотеки

- Ядро определяет глобальный объект BX, который содержит статические методы для работы с событиями, DOM и др.

- `core_*.js` – файлы расширений
  - `core_ajax.js` – работа с Ajax-запросами
  - `core_date.js` – форматирование даты



1С·БИТРИКС

Подключение в PHP

- `CUtil::InitJSCore([array $arExtensions])`

  Подключает ядро, стили и языковые сообщения библиотеки
  `$arExtensions` – массив требуемых расширений библиотеки

- Пример:

```
<?
   CUtil::InitJSCore(Array("ajax", "window"));
  ?>
```

1С·БИТРИКС

Регистрация своих расширений

```
<?
 CJSCore::RegisterExt("my_extension", Array(
  "js" => "/path/to/js/my_ext.js",
  "css" => "/path/to/css/my_ext.css",
  "lang" => "/path/to/lang/".LANGUAGE_ID. "/lang.php",
  "rel" => Array("ajax", "timer", "popup", "ls")
 ));
?>
```

1С·БИТРИКС

Начало работы

- BX.ready(function(){

```
  //свой код
});
```

- BX.ready() гарантирует, что DOM-дерево доступно для изменения

Манипуляции с DOM-объектами

- DOMNode **BX.create**(**tag**[, **data**[, **context**]])

  Создает узел DOM-дерева с именем тега tag.
  Описательный объект data может иметь следующие поля:

```
{
  tag: имя тега узла,
  props: { дополнительные js-свойства},
  style: { стили узла },
  events: {обработчики событий узла},
  children: [массив дочерних узлов],
  text: текстовое содержимое узла,
  html: HTML-содержимое узла
 }
```

- DOMNode **BX.adjust**(DOMNode **node**, object **data**)

  Изменяет свойства узла node.

Манипуляции с DOM-объектами

- DOMNode **BX.addClass**(DOMNode **node**, string **className**)

  Добавляет узлу node CSS-класс className, если он такового не имеет.

- DOMNode **BX.removeClass**(DOMNode **node**, string **className**)

  Удаляет CSS-класс className, у узла node.

- DOMNode **BX.toggleClass**(DOMNode **node**, string|array **className**)

  Переключить наличие/отсутствие CSS-класса className у узла node или устроить ротацию CSS-классов, если className – массив.

- bool **BX.hasClass**(DOMNode **node**, string **className**)

  Проверяет, есть ли уже у узла node CSS-класс className.

Манипуляции с DOM-объектами

- DOMNode **BX. style**(DOMNode **node**, string **property**[, string **value**])

  Получить текущее значение стиля property узла node или установить его в значение value.

- null **BX.remove**(DOMNode **node**)

  Удалить узел DOM-структуры.

- DOMNode **BX.cleanNode**(DOMNode **node**[, bool **bSuicide**])

  Очистить DOM-узел от всех дочерних элементов. Если второй параметр равен true, сам узел также будет удален.

- `BX.show`(DOMNode **node**), `BX.hide`(DOMNode **node**)

  Показать/скрыть узел.

Поиск элементов в DOM

- `Array BX.findChildren(DOMNode obj, Object params, bool recursive)`
  `DOMNode BX.findChild(DOMNode obj, Object params, bool recursive)`

  Поиск потомков

  params : {

  tagName|**tag**: *имя тега требуемого узла*,

  className|**class**: *CSS-класс, который должен содержать требуемый узел,*

  attribute: {attribute: value, attribute: value} | attribute | [attribute, attribute], - *атрибут или коллекция атрибутов, которые должны присутствовать в требуемом узле*

  property: { prop: value, prop: value} | prop | [prop, prop] – *свойство или коллекция свойств, которые должны присутствовать в требуемом узле*

  callback: функция для произвольной фильтрации

  }

Поиск элементов в DOM

- `DOMNode BX(String id)`

  Поиск элемента по id

- `DOMNode BX.findParent(DOMNode obj, Object params)`

  Поиск родителя

- `DOMNode BX.findNextSibling(DOMNode obj, Object params)`
  `DOMNode BX.findPreviousSibling(DOMNode obj, Object params)`

  Поиск соседних элементов

1С·БИТРИКС

Работа с событиями

- void **BX.bind**(DOMNode **element**, String **event**, Function **handler**)

  Установить функцию handler в качестве обработчика события event элемента element.

  Пример: BX.bind(BX("test"), "click", function() {alert(1); })

- void **BX.unbind**(DOMNode **element**, String **event**, Function **handler**)

  Снять обработчик handler события event элемента element.

- void **BX.unbindAll**(DOMNode **element**)

  Снять все зарегистрированные обработчики событий с элемента element

- **BX.eventCancelBubble**(Event **event**)

  Запретить всплывание события

- **BX.eventReturnFalse**(Event **event**)

  ...ть действие по умолчанию

Пользовательские событиями

- **BX.addCustomEvent**(Object **eventObject**, string **eventName**, Function **handler**)

  Назначить обработчик eventHandler кастомному событию с именем eventName, возникающем в объекте eventObject.

- **BX.removeCustomEvent**(Object **eventObject**, string **eventName**, Function **handler**)

  Удалить обработчик eventHandler кастомного события с именем eventName.

- **BX.onCustomEvent**(Object **eventObject**, string **eventName**[, Array **arEventParams**])

  Вызвать все обработчики события eventName для объекта eventObject и передать перевым параметром arEventParams.

Размеры окна и координаты

- object BX.pos(DOMNode node)
  Возвращает координаты и размеры узла, объект с ключами top, right, bottom, left, width, height.

- object BX.GetWindowScrollSize([DOMDocument doc])
  Возвращает размеры скролла, объект с ключами scrollWidth, scrollHeight

- object BX.GetWindowScrollPos([DOMDocument doc])
  Возвращает позицию скролла, объект с ключами scrollLeft, scrollTop

- object BX.GetWindowInnerSize([DOMDocument doc])
  Возвращает размер окна, объект с ключами innerWidth, erHeight

Утилиты

- BX.util – php-аналоги
  - BX.util.array_merge()
  - BX.util.array_unique()
  - BX.util.in_array()
  - BX.util.trim()
  - BX.util.htmlspecialchars()
  - BX.util.urlencode()
- BX.browser – проверка на браузер
  - BX.browser.isIE(), BX.browser.isOpera(), BX.browser.isSafari()
- BX.type – проверка типа объекта
  - BX.type.isArray(), BX.type.isDate() и др.
- Демонстрация

1С·БИТРИКС

Работа с LocalStorage (core_ls.js)

- Поддерживается всеми современными браузерами, а также IE7+
- Применение
  - Синхронизация данных между окнами браузера
  - Кеширование данных на клиенте
  - Оптимизация ajax-запросов
- Демонстрация

Работа с LocalStorage (core_ls.js)

- **BX.localStorage.set(key, value, ttl)**

  Устанавливает значение записи равным value с ключом key . ttl – время жизни в секундах.

- **BX.localStorage.get(key)**

  Получить значение по ключу key

- **BX.localStorage.remove(key)**

  Удалить запись с ключом key

- События
  - onLocalStorageSet – установка значения в LS
  - onLocalStorageRemove – удаление записи в LS
  - onLocalStorageChange – изменение значение в LS

Ajax-расширение core_ajax.js

- XMLHttpRequest **BX.ajax**(object **params**)
  Низкоуровневая функция для отправки ajax-запросов.

```
params:
{
 url: URL запроса
 method: GET|POST
 dataType: html|json|script – данные какого типа предполагаются в ответе
 timeout: 60 – таймаут запроса в секундах
 async: true|false – должен ли запрос быть асинхронным или нет
 processData: true|false – нужно ли сразу обрабатывать данные?
 scriptsRunFirst: false|true – нужно ли выполнять все найденные скрипты
                  перед тем, как отдавать содержимое обработчику
    start: true|false – отправить ли запрос сразу или он будет запущен
вручную
 onsuccess: функция-обработчик результата
 onfailure: функция-обработчик ошибки
}
```

1С·БИТРИКС

Ajax-расширение core_ajax.js

- XHRequest **BX.ajax.get**(string **url**, function **callback**)

  Отправка GET-запроса и передача результата обработчику callback.

- XHRequest **BX.ajax.post**(string **url**, string|object **data**, function **callback**)

  Отправка POST-запроса и передача результата обработчику callback. Параметр data – это строка или ассоциативный массив POST-данных запроса.

- XHRequest **BX.ajax.insertToNode**(string **url**, string|DOMNode **node**)

  Запросить url и вставить результат в контейнер node. Если node – строка, то параметр будет интерпретирован как идентификатор контейнера.

- XMLHttpRequest **BX.ajax.loadJSON**(string **url**, function **callback**)

  Загрузить json-объект из заданного url и передать его обработчику callback

Пользовательские настройки

- **BX. userOptions.save**(**sCategory**, **sName**, **sValName**, **sVal**, **bCommon**)
  Сохранение значения пользовательской настройки.

- **BX.userOptions.del** (**sCategory**, **sName**, **bCommon**, **callback**) –
  Удаление значения пользовательской настройки.

1С·БИТРИКС

Анимация

Class **BX.fx**(Object **options**)

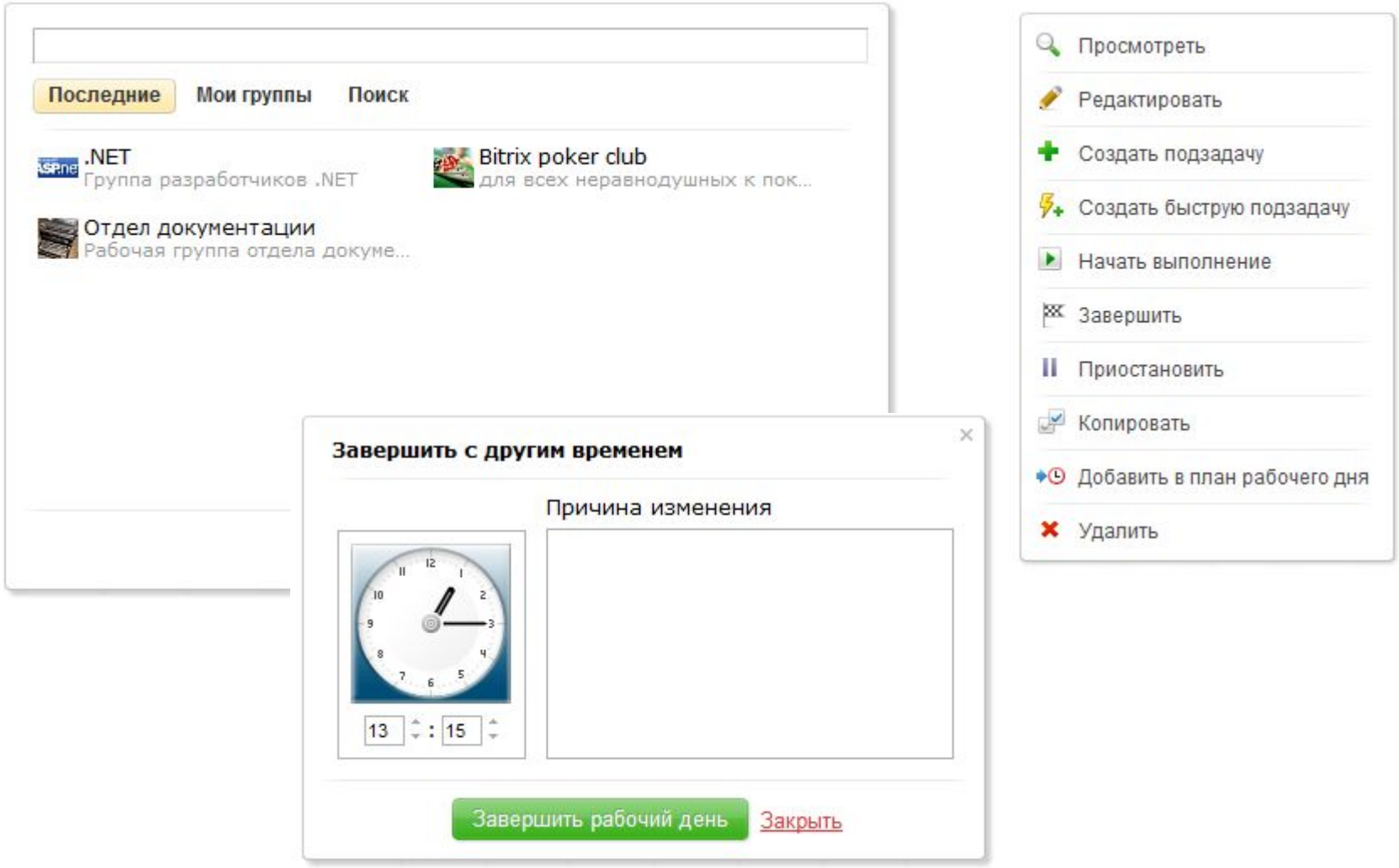
```
options: {
   start: стартовое значение и набор значений
   finish: конечно значение и набор значений
   time: время, в течение которого должна производиться анимация
   type: linear|accelerated|decelerated – закон изменения параметров
   callback : обработчик значения параметра
   callback_start : обработчик начала анимации
   callback_complete : обработчик завершения анимации
   step: длительность промежутка анимации
}
```

BX.fx.prototype.start() – начать анимацию
BX.fx.prototype.pause() – затормозить/запустить снова анимацию
BX.fx.prototype.stop() – закончить анимацию

Объект класса BX.fx вызывает обработчик **callback** через определенные промежутки времени, задаваемые параметром **step**, в течение периода **time**.

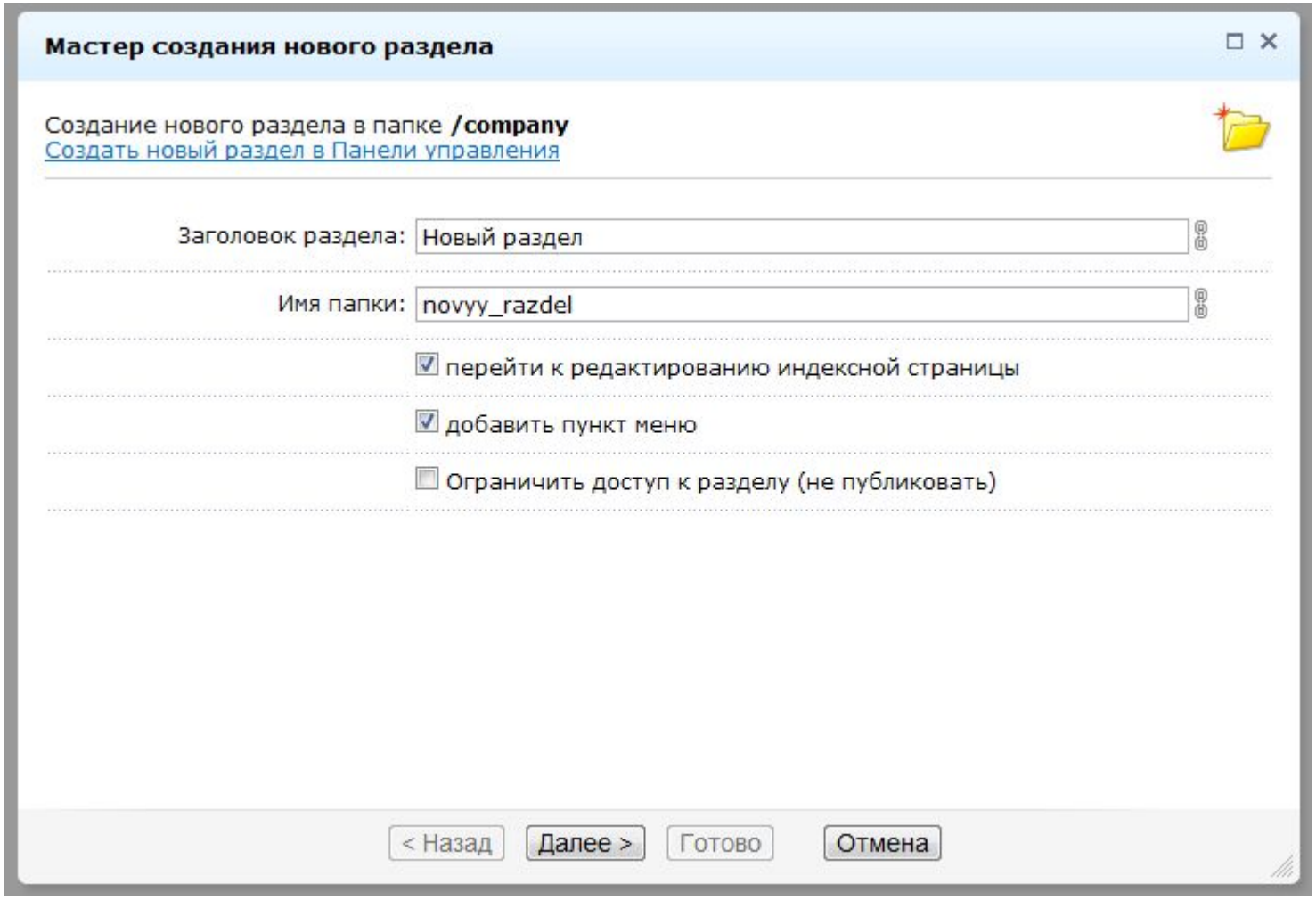
1С·БИТРИКС

Оконная библиотека (core_popup.js)



1С·БИТРИКС

Оконная библиотека (core_window.js)



1С·БИТРИКС

Оконная библиотека

- `Class BX.PopupWindow(id, bindElement, options)`
- `Class BX.CDialog(params)`
- `Class BX.CAdminDialog(params)`

- Демонстрация

Форматирование даты (core_date.js)

- `BX.date.format(format, [date[, currentDate[, isUtc]]])`

  format – полный аналог формата php-функции date, за исключением формата "T" и "e" (символьное название таймзоны).

  date – дата в формате timestamp или объект класс Date.

  currentDate – текущая дата (timestamp|Date).

  isUTC – дата в UTC? Если необходимо работать с датами в UTC.

- ```
  BX.date.format("d-m-Y H:i:s") // 24-01-2012 18:22:12
  BX.date.format("j F Y H:i:s") //  24 Января 2012 18:22:32
  BX.date.format("iago", new Date(2012, 0, 23)) // 2545 минут назад
  ```
  ```

Документация будет!

1С·БИТРИКС

Заключение

- JS-библиотека не накладывает никаких ограничений - используйте совместно свой любимый фреймворк.
- jQuery включена в продукт для удобства разработки дополнений из MarketPlace.
- ```
  <?
     CUtil::InitJSCore(Array("jquery"));
  ?>
  ```

1С·БИТРИКС
  ```

Спасибо за внимание!
Вопросы?

- E-mail: co@bitrix.ru
- Twitter: twitter.com/compote

1С·БИТРИКС