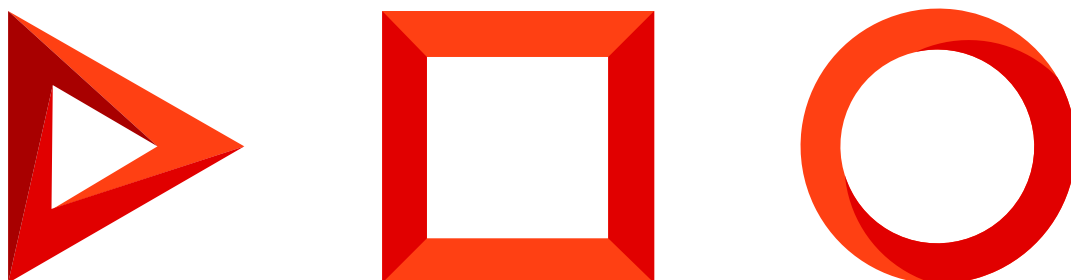


# Бизнес-правила мобильного приложения

Версия 8.0



Эта документация предоставляется с ограничениями на использование и защищена законами об интеллектуальной собственности. За исключением случаев, прямо разрешенных в вашем лицензионном соглашении или разрешенных законом, вы не можете использовать, копировать, воспроизводить, переводить, транслировать, изменять, лицензировать, передавать, распространять, демонстрировать, выполнять, публиковать или отображать любую часть в любой форме или посредством любые значения. Обратный инжиниринг, дизассемблирование или декомпиляция этой документации, если это не требуется по закону для взаимодействия, запрещены.

Информация, содержащаяся в данном документе, может быть изменена без предварительного уведомления и не может гарантировать отсутствие ошибок. Если вы обнаружите какие-либо ошибки, сообщите нам о них в письменной форме.

# Содержание

<b>Бизнес-правила мобильного приложения</b>	<b>4</b>
<b>Выполнить фильтрацию</b>	<b>4</b>
Пример 1	4
Пример 2	5
Пример 3	5
<b>Выделить поле по условию</b>	<b>6</b>
Реализация примера	7
<b>Сбросить отрицательные значения в 0</b>	<b>7</b>
Реализация примера	8
<b>Сгенерировать заголовок активности</b>	<b>8</b>
Реализация примера	8
<b>Свойства объекта config</b>	<b>9</b>
Базовые бизнес-правила	9
Бизнес-правило Обязательность заполнения (Terrasoft.RuleTypes.Requirement)	10
Бизнес-правило Видимость (Terrasoft.RuleTypes.Visibility)	11
Бизнес-правило Доступность (Terrasoft.RuleTypes.Activation)	12
Бизнес-правило Фильтрация (Terrasoft.RuleTypes.Filtration)	13
Бизнес-правило Взаимная фильтрация (Terrasoft.RuleTypes.MutualFiltration)	14
Бизнес-правило Регулярное выражение (Terrasoft.RuleTypes.RegExp)	15
Пользовательские бизнес-правила	16

# Бизнес-правила мобильного приложения



**Бизнес-правила** — это один из механизмов Creatio, позволяющий настраивать поведение полей на странице записи. С их помощью можно, например, настроить обязательность и видимость полей, их доступность и т. п.

**Важно.** Бизнес-правила работают только на страницах записи и просмотра записей.

Добавление бизнес-правила на страницу выполняется с помощью метода

`Terrasoft.sdk.Model.addBusinessRule(name, config)`, где

- `name` — название модели, связанной со страницей записи, например, "Contact".
- `config` — объект, определяющий свойства бизнес-правила. Перечень свойств зависит от конкретного типа бизнес-правила.

В мобильном приложении существует возможность добавлять бизнес-правило, реализующее пользовательскую логику — пользовательское бизнес-правило. Для такого бизнес-правила предусмотрен тип `Terrasoft.RuleTypes.Custom`.

## Выполнить фильтрацию



### Пример 1

**Пример.** Отфильтровать значения в колонке по условию.

На детали [ *Продукты в счете* ] при выборе значения из справочной колонки [ *Продукт* ] доступны только те продукты, у которых колонка [ *Active* ] содержит значение `true`.

### Реализация примера

#### Пример фильтрации

```
Terrasoft.sdk.Model.addBusinessRule("InvoiceProduct", {
  ruleType: Terrasoft.RuleTypes.Filtration,
  events: [Terrasoft.BusinessRuleEvents.Load],
```

```

    triggeredByColumns: ["Product"],
    filters: Ext.create("Terrasoft.Filter", {
        modelName: "Product",
        property: "Active",
        value: true
    })
});

```

## Пример 2

**Пример.** Отфильтровать значения в колонке по значению другой колонки.

На странице записи раздела [ Счета ], поле [ Контакт ] должно фильтроваться на основании значения в поле [ Контрагент ].

### Реализация примера

#### Пример фильтрации

```

Terrasoft.sdk.Model.addBusinessRule("Invoice", {
    ruleType: Terrasoft.RuleTypes.Filtration,
    events: [Terrasoft.BusinessRuleEvents.Load, Terrasoft.BusinessRuleEvents.ValueChanged],
    triggeredByColumns: ["Account"],
    filteredColumn: "Contact",
    filters: Ext.create("Terrasoft.Filter", {
        property: "Account"
    })
});

```

## Пример 3

**Пример.** Добавить и удалить фильтрацию по пользовательской логике.

### Реализация примера

#### Пример фильтрации

```

Terrasoft.sdk.Model.addBusinessRule("Activity", {
    name: "ActivityResultByAllowedResultFilterRule",
    position: 1,

```

```

ruleType: Terrasoft.RuleTypes.Custom,
triggeredByColumns: ["Result"],
events: [Terrasoft.BusinessRuleEvents.ValueChanged, Terrasoft.BusinessRuleEvents.Load],
executeFn: function(record, rule, column, customData, callbackConfig) {
    var allowedResult = record.get("AllowedResult");
    var filterName = "ActivityResultByAllowedResultFilter";
    if (!Ext.isEmpty(allowedResult)) {
        var allowedResultIds = Ext.JSON.decode(allowedResult, true);
        var resultIdsAreCorrect = true;
        for (var i = 0, ln = allowedResultIds.length; i < ln; i++) {
            var item = allowedResultIds[i];
            if (!Terrasoft.util.isGuid(item)) {
                resultIdsAreCorrect = false;
                break;
            }
        }
        if (resultIdsAreCorrect) {
            var filter = Ext.create("Terrasoft.Filter", {
                name: filterName,
                property: "Id",
                funcType: Terrasoft.FilterFunctions.In,
                funcArgs: allowedResultIds
            });
            record.changeProperty("Result", {
                addFilter: filter
            });
        } else {
            record.changeProperty("Result", {
                removeFilter: filterName
            });
        }
    } else {
        record.changeProperty("Result", {
            removeFilter: filterName
        });
    }
    Ext.callback(callbackConfig.success, callbackConfig.scope, [true]);
}
});

```

## Выделить поле по условию



Сложный

**Пример.** Требуется выделить поле с результатом активности, если ее статус "Завершена", само поле [ *Результат* ] не заполнено и есть значение в колонке [ *ProcessElementId* ].

## Реализация примера

### Выделение поля по условию

```
// Правило для страницы активности.
Terrasoft.sdk.Model.addBusinessRule("Activity", {
  // Название бизнес-правила.
  name: "ActivityResultRequiredByStatusFinishedAndProcessElementId",
  // Тип бизнес-правила: пользовательское.
  ruleType: Terrasoft.RuleTypes.Custom,
  // Правило инициируется колонками Status и Result.
  triggeredByColumns: ["Status", "Result"],
  // Правило отработает перед сохранением данных и после изменения данных.
  events: [Terrasoft.BusinessRuleEvents.ValueChanged, Terrasoft.BusinessRuleEvents.Save],
  // Функция-обработчик.
  executeFn: function(record, rule, column, customData, callbackConfig) {
    // Признак корректности свойства и правила.
    var isValid = true;
    // Значение колонки ProcessElementId.
    var processElementId = record.get("ProcessElementId");
    // Если значение не пустое.
    if (processElementId && processElementId !== Terrasoft.GUID_EMPTY) {
      // Установка признака корректности.
      isValid = !(record.get("Status.Id") === Terrasoft.Configuration.ActivityStatus.Finished
        Ext.isEmpty(record.get("Result")));
    }
    // Изменение свойств колонки Result.
    record.changeProperty("Result", {
      // Установка признака корректности колонки.
      isValid: {
        value: isValid,
        message: Terrasoft.LS["Sys.RequirementRule.message"]
      }
    });
    // Асинхронный возврат значений.
    Ext.callback(callbackConfig.success, callbackConfig.scope, [isValid]);
  }
});
```

## Сбросить отрицательные значения в 0



**Пример.** Реализовать логику сбрасывания отрицательных значений в 0.

## Реализация примера

### Сброс отрицательных значений в 0

```
Terrasoft.sdk.Model.addBusinessRule("Opportunity", {
  name: "OpportunityAmountValidatorRule",
  ruleType: Terrasoft.RuleTypes.Custom,
  triggeredByColumns: ["Amount"],
  events: [Terrasoft.BusinessRuleEvents.ValueChanged, Terrasoft.BusinessRuleEvents.Save],
  executeFn: function(model, rule, column, customData, callbackConfig) {
    var revenue = model.get("Amount");
    if ((revenue < 0) || Ext.isEmpty(revenue)) {
      model.set("Amount", 0, true);
    }
    Ext.callback(callbackConfig.success, callbackConfig.scope);
  }
});
```

## Сгенерировать заголовок активности



**Пример.** Реализовать генерацию заголовка активности для решения FieldForce.

## Реализация примера

### Генерация заголовка активности

```
Terrasoft.sdk.Model.addBusinessRule("Activity", {
  name: "FieldForceActivityTitleRule",
  ruleType: Terrasoft.RuleTypes.Custom,
  triggeredByColumns: ["Account", "Type"],
  events: [Terrasoft.BusinessRuleEvents.ValueChanged, Terrasoft.BusinessRuleEvents.Load],
  executeFn: function(record, rule, column, customData, callbackConfig, event) {
    if (event === Terrasoft.BusinessRuleEvents.ValueChanged || record.phantom) {
      var type = record.get("Type");
      var typeId = type ? type.get("Id") : null;
      if (typeId !== Terrasoft.Configuration.ActivityTypes.Visit) {
        Ext.callback(callbackConfig.success, callbackConfig.scope, [true]);
      }
    }
  }
});
```



```

        return;
    }
    var account = record.get("Account");
    var accountName = (account) ? account.getPrimaryDisplayColumnValue() : "";
    var title = Ext.String.format("{0}: {1}", Terrasoft.LocalizableStrings.FieldForceTit
    record.set("Title", title, true);
}
Ext.callback(callbackConfig.success, callbackConfig.scope, [true]);
}
});

```

## Свойства объекта config C#



### Базовые бизнес-правила

Базовое бизнес-правило является абстрактным классом, т.е. все бизнес-правила должны быть его наследниками.

Свойства конфигурационного объекта `config` могут быть использованы наследниками базового бизнес-правила.

### Свойства конфигурационного объекта config

`ruleType`

Тип правила. Значение должно входить в перечисление `Terrasoft.RuleTypes`.

`triggeredByColumns`

Массив колонок, инициирующих срабатывание бизнес-правила.

`message`

Текстовое сообщение, которое выводится под элементом управления, который связан с колонкой, в случае невыполнения бизнес-правила. Необходимо для правил, сообщающих пользователю предупреждающую информацию.

`name`

Уникальное имя бизнес-правила. Необходимо, если нужно удалить правило методами `Terrasoft.sdk`.

`position`

Позиция бизнес-правила, определяющая приоритет вызова правила в очереди текущих запускаемых правил.

events

Массив событий, определяющий время запуска бизнес-правил. Должен содержать значения, входящие в перечисление `Terrasoft.BusinessRuleEvents`.

Возможные значения ( `Terrasoft.BusinessRuleEvents` )

Save	Правило отработает перед сохранением данных.
ValueChanged	Правило отработает после изменения данных (при редактировании).
Load	Правило отработает при открытии страницы записи.

## Бизнес-правило [ Обязательность заполнения ] (`Terrasoft.RuleTypes.Requirement`) C#

Определяет обязательность заполнения поля на странице записи.

### Свойства конфигурационного объекта config

ruleType

Для этого правила должно содержать значение `Terrasoft.RuleTypes.Requirement`.

requireType

Тип проверки. Значение должно входить в перечисление `Terrasoft.RequirementTypes`. Правило может проверять одну или все колонки из `triggeredByColumns`.

triggeredByColumns

Массив колонок, инициирующих срабатывание бизнес-правила. Если тип проверки равен `Terrasoft.RequirementTypes.Simple`, то должна быть указана одна колонка в массиве.

Возможные значения ( `Terrasoft.RequirementTypes` )

Simple	Проверка значения в одной колонке.
OneOf	Одна из колонок, указанных в <code>triggeredByColumns</code> должна быть обязательно заполнена.

**Пример использования**

```
Terrasoft.sdk.Model.AddBusinessRule("Contact", {
    ruleType: Terrasoft.RuleTypes.Requirement,
    requireType : Terrasoft.RequirementTypes.OneOf,
    events: [Terrasoft.BusinessRuleEvents.Save],
    triggeredByColumns: ["HomeNumber", "BusinessNumber"],
    columnNames: ["HomeNumber", "BusinessNumber"]
});
```

## Бизнес-правило [ Видимость ] (Terrasoft.RuleTypes.Visibility) C#

С помощью этого бизнес-правила можно скрывать и отображать поля по определенному условию.

### Свойства конфигурационного объекта config

#### ruleType

Для этого правила должно содержать значение `Terrasoft.RuleTypes.Visibility`.

#### triggeredByColumns

Массив колонок, инициирующих срабатывание бизнес-правила.

#### events

Массив событий, определяющий время запуска бизнес-правил. Должен содержать значения, входящие в перечисление `Terrasoft.BusinessRuleEvents`.

#### conditionalColumns

Массив условий срабатывания бизнес-правила. Обычно это определенные значения колонок.

#### dependentColumnNames

Массив названий колонок, к которым применяется данное бизнес-правило.

**Пример использования**

```
Terrasoft.sdk.Model.AddBusinessRule("Account", {
    ruleType: Terrasoft.RuleTypes.Visibility,
```

```
conditionalColumns: [
    {name: "Type", value: Terrasoft.Configuration.Consts.AccountTypePharmacy}
],
triggeredByColumns: ["Type"],
dependentColumnNames: ["IsRx", "IsOTC"]
});
```

Поля, связанные с колонками `IsRx` и `IsOTC` будут отображены, если колонка `Type` будет содержать значение, определенное константой `Terrasoft.Configuration.Consts.AccountTypePharmacy`.

```
Terrasoft.Configuration.Consts = {
    AccountTypePharmacy: "d12dc11d-8c74-46b7-9198-5a4385428f9a"
};
```

Вместо константы можно использовать значение "d12dc11d-8c74-46b7-9198-5a4385428f9a".

## Бизнес-правило [ *Доступность* ] (Terrasoft.RuleTypes.Activation) C#

С помощью этого бизнес-правила можно делать поля недоступными (или наоборот — доступными) для ввода значений по определенному условию.

### Свойства конфигурационного объекта config

ruleType

Для этого правила должно содержать значение `Terrasoft.RuleTypes.Activation`.

triggeredByColumns

Массив колонок, инициирующих срабатывание бизнес-правила.

events

Массив событий, определяющий время запуска бизнес-правил. Должен содержать значения, входящие в перечисление `Terrasoft.BusinessRuleEvents`.

conditionalColumns

Массив условий срабатывания бизнес-правила. Обычно это определенные значения колонок.

dependentColumnNames

Массив названий колонок, к которым применяется данное бизнес-правило.

Доступность поля, связанного с колонкой `Stock`, зависит от значения в колонке `IsPresence`.

#### Пример использования

```
Terrasoft.sdk.Model.addBusinessRule("ActivitySKU", {
  ruleType: Terrasoft.RuleTypes.Activation,
  events: [Terrasoft.BusinessRuleEvents.Load, Terrasoft.BusinessRuleEvents.ValueChanged],
  triggeredByColumns: ["IsPresence"],
  conditionalColumns: [
    {name: "IsPresence", value: true}
  ],
  dependentColumnNames: ["Stock"]
});
```

## Бизнес-правило [ Фильтрация ] (Terrasoft.RuleTypes.Filtration)

Это бизнес-правило можно использовать для фильтрации значений справочных колонок по условию, либо по значению другой колонки.

### Свойства конфигурационного объекта config

ruleType

Для этого правила должно содержать значение `Terrasoft.RuleTypes.Filtration`.

triggeredByColumns

Массив колонок, инициирующих срабатывание бизнес-правила.

events

Массив событий, определяющий время запуска бизнес-правил. Должен содержать значения, входящие в перечисление `Terrasoft.BusinessRuleEvents`.

filters

Фильтр. Свойство должно содержать экземпляр класса `Terrasoft.Filter`.

filteredColumn

Колонка, на основании которой выполняется фильтрация значений.

## Бизнес-правило [ *Взаимная фильтрация* ] (Terrasoft.RuleTypes.MutualFiltration) C#

Это бизнес-правило позволяет выполнять взаимную фильтрацию двух справочных полей. Работает только с колонками, связанными отношением "один-ко-многим", например, [ Страна ] — [ Город ]. Для каждой связки полей необходимо создавать свое бизнес-правило. Например, для связок [ Страна ] — [ Область ] — [ Город ] и [ Страна ] — [ Город ] необходимо создать три бизнес-правила:

- [ Страна ] — [ Область ];
- [ Область ] — [ Город ];
- [ Страна ] — [ Город ].

## Свойства конфигурационного объекта config

ruleType

Для этого правила должно содержать значение `Terrasoft.RuleTypes.MutualFiltration`.

triggeredByColumns

Массив колонок, инициирующих срабатывание бизнес-правила.

connections

Массив объектов, конфигурирующих отношение связей.

Взаимная фильтрация полей [Страна], [Область] и [Город]

```
Terrasoft.sdk.Model.addBusinessRule("ContactAddress", {
  ruleType: Terrasoft.RuleTypes.MutualFiltration,
  triggeredByColumns: ["City", "Region", "Country"],
  connections: [
    {
      parent: "Country",
      child: "City"
    },
    {
      parent: "Country",
      child: "Region"
    },
    {
```

```

        parent: "Region",
        child: "City"
    }
]
});

```

Взаимная фильтрация полей [Контакт], [Контрагент]

```

Terrasoft.sdk.Model.addBusinessRule("Activity", {
    ruleType: Terrasoft.RuleTypes.MutualFiltration,
    triggeredByColumns: ["Contact", "Account"],
    connections: [
        {
            parent: "Contact",
            child: "Account",
            connectedBy: "PrimaryContact"
        }
    ]
});

```

## Бизнес-правило [ *Регулярное выражение* ] (Terrasoft.RuleTypes.RegExp) C#

Выполняет проверку соответствия значения колонки регулярному выражению.

### Свойства конфигурационного объекта config

ruleType

Для этого правила должно содержать значение `Terrasoft.RuleTypes.RegExp`.

RegExp

Регулярное выражение, на соответствие которому проверяются все колонки из массива `triggeredByColumns`.

triggeredByColumns

Массив колонок, инициирующих срабатывание бизнес-правила.

#### Пример использования

```
Terrasoft.sdk.Model.addBusinessRule("Contact", {
  ruleType: Terrasoft.RuleTypes.RegExp,
  regExp : /^[0-9\(\)\ \/\+ \-]*$/
  triggeredByColumns: ["HomeNumber", "BusinessNumber"]
});
```

## Пользовательские бизнес-правила

При добавлении пользовательского бизнес-правила с помощью метода

`Terrasoft.sdk.Model.addBusinessRule(name, config)` можно использовать свойства конфигурационного объекта `config` базового бизнес-правила. Также дополнительно предусмотрено свойство `executeFn`.

### Свойства конфигурационного объекта config

`ruleType`

Тип правила. Для пользовательских правил должно содержать значение `Terrasoft.RuleTypes.Custom`.

`triggeredByColumns`

Массив колонок, инициирующих срабатывание бизнес-правила.

`events`

Массив событий, определяющий время запуска бизнес-правил. Должен содержать значения из перечисления `Terrasoft.BusinessRuleEvents`. Значение по умолчанию:

`Terrasoft.BusinessRuleEvents.ValueChanged`.

**Возможные значения** (`Terrasoft.BusinessRuleEvents`)

Save	Правило сработает перед сохранением данных.
ValueChanged	Правило сработает после изменения данных (при редактировании).
Load	Правило сработает при открытии страницы записи.

`executeFn`

Функция-обработчик, содержащая пользовательскую логику выполнения бизнес-правила.

### Свойства функции-обработчика executeFn

Функция-обработчик объявляется в свойстве `executeFn`.



## Сигнатура функции-обработчика

```
executeFn: function(record, rule, checkColumnName, customData, callbackConfig, event) { }
```

### Параметры

record	Запись, для которой выполняется бизнес-правило.
rule	Экземпляр текущего бизнес-правила.
checkColumnName	Название колонки, которая вызвала срабатывание бизнес-правила.
customData	Объект, разделяемый между всем правилами. Не используется. Оставлен для совместимости с предыдущими версиями.
callbackConfig	Конфигурационный объект асинхронного возврата <code>Ext.callback</code> .
event	Событие, по которому было запущено бизнес-правило.

При завершении работы функции необходимо обязательно вызвать или `callbackConfig.success`, или `callbackConfig.failure`.

### Варианты вызова

```
Ext.callback(callbackConfig.success, callbackConfig.scope, [result]);
Ext.callback(callbackConfig.failure, callbackConfig.scope, [exception]);
```

Где:

- `result` — возвращаемое логическое значение, полученное при выполнении функции (`true` / `false`).
- `exception` — исключение типа `Terrasoft.Exception`, возникшее в функции-обработчике.

## Методы

В исходном коде функции-обработчика удобно использовать следующие методы модели, передаваемой в параметре `record`:

---

```
get(columnName)
```

Для получения значения колонки записи. Аргумент `columnName` должен содержать название колонки.

---

```
set(columnName, value, fireEventConfig)
```

Для установки значения колонки записи.

### Параметры

columnName	Название колонки.
value	Значение, присваиваемое колонке.
fireEventConfig	Конфигурационный объект для установки свойств, передаваемых в событие изменения колонки.

`changeProperty(columnName, propertyConfig)`

Для изменения свойств колонки, кроме ее значения. Аргумент `columnName` должен содержать название колонки, а `propertyConfig` — объект, устанавливающий свойства колонки.

### Свойства объекта `propertyConfig`

disabled	Активность колонки. Если <code>true</code> , то элемент управления, связанный с колонкой, будет неактивным и закрытым для работы.
readOnly	Признак "только для чтения". Если <code>true</code> , то элемент управления, связанный с колонкой, будет доступен только для чтения. Если <code>false</code> — для чтения и записи.
hidden	Видимость колонки. Если <code>true</code> , то элемент управления, связанный с колонкой, будет скрыт. Если <code>false</code> — отображен.
addFilter	Добавить фильтр. Если свойство указано, то в нем должен быть указан фильтр типа <code>Terrasoft.Filter</code> , который будет добавлен к фильтрации колонки. Свойство используется только для справочных полей.
removeFilter	Удалить фильтр. Если свойство указано, то в нем должно быть указано имя фильтра, который будет удален из фильтрации колонки. Свойство используется только для справочных полей.
isValid	Признак корректности колонки. Если свойство указано, то оно изменит признак корректности элемента управления, связанного с колонкой. Если колонка некорректна, то это может означать отмену событий по сохранению записи, а также может привести к признанию записи некорректной.

### Пример изменения свойств (но не значения) колонки `Owner`

```
record.changeProperty("Owner", {
    disabled: false,
    readOnly: false,
    hidden: false,
    addFilter: {
        property: "IsChief",
        value: true
    },
    isValid: {
        value: false,
        message: LocalizableStrings["Owner_should_be_a_chief_only"]
    }
});
```