

Обрешетка 2.0

Проблематика базовая

Текущая реализация обрешётки предполагает простой набор условий. После начала использования к механизму добавилось ряд требований, которые невозможно покрыть не изменив подход:

- возможность менять шаг обрешётки;
- сделать расходники регионально-зависимыми;
- дать возможность менять формулы расходников;
- дать возможность заводить расходники, зависящие от условий (от толщины доски, от ширины доски, от сорта доски, от материала, от профиля, ...);
- дать возможность заводить вариации расходников (чтобы для уголка в обрешётке можно было выбрать вариацию: как правило, более дорогой или более дешёвый), вариации саморезов;
- реализовать возможность обрабатывать деревянную составляющую обрешётки (например, антисептировать брусok обрешётки);
- реализовать админку для управления массивом условий и расходников.

Разработана новая концепция на уровне БД, реализуем новый фронт и админку

Проблематика сопроводительная

У нас джсон, возвращаемый на разных калькуляторах, содержит полный набор данных. То есть в калькуляторе обрешётки есть данные для калькулятора покраски фасадов и наоборот.

Нужно разгрузить эту ситуацию и сделать более управляемой. Для этого в публичной части мы сделаем так, чтобы каждый компонент знал о том, какие данные нужны ему для работы. Общий компонент перед запросом джсона "собирает заявки" с тех компонентов, которые входят в приложение, формирует общий уникальный список (потому что многие пункты в заявках будут повторяться) и подаёт его в запросе на джсон. По сути это список ключей из innerData, который каждый из компонентов использует при инициализации data.

Бэк реагирует на заявку и комплектует джсон теми данными, которые были запрошены.

Концепции

Расходники

В бд на бэке хранение производится в линейном виде:

- стена + камень + вариация 1 + саморез 1 + условие 1 + формула 1
- стена + камень + вариация 2 + саморез 2 + условие 1 + формула 1
- стена + камень + вариация 2 + саморез 1 + условие 2 + формула 1
- стена + камень + вариация 2 + саморез 2 + условие 2 + формула 1

В бд есть ветвление на:

- расходник + вариация расходника
- саморез + вариация самореза.

...где **расходник** и **саморез** — логическая группировка чтобы общие параметры не дублировать (в основном лингвистика и параметры типа "продаём штуками, погонными метрами, упаковками").

На фронте такого ветвления нет, здесь мы оперируем просто терминами "расходник" и "саморез".

Кроме того, джсон представляет собой не дерево а набор комплектующих. То есть если для поверхности (камень, газобетон, ...) предусмотрено например пять вариаций уголков, это будет не дерево сгруппированное по уголку с пятью вариациями а пять разных объектов-уголков. Обусловлено тем, что мы вводим понятие *условий* (или *правил применения*), когда один и тот же расходник (но разные его вариации) отображаются в зависимости от условий. Усложняется всё вариациями (как самих расходников так и саморезов под ним): например, для одного и того же уголка могут быть доступны разные саморезы на выбор. Можно было бы класть их в ту же ветку, что и уголок, в виде под-дерева. Но возможна и другая ситуация: у уголка три типа саморезов. *Два* отвечают **Условию1** (и в таком случае на странице пользователь выбирает, какой из саморезов заказать, а *третий* — **Условию2** (например, он предназначен для более толстой доски). Третий саморез не может быть размещённым в тот же джсон что и первые два самореза и будет идти отдельной веткой. А значит, решив мы сгруппировать первые два самореза а третий дать отдельно, мы нагрузку на фронт не снижаем, но усложняем анализ в целом.

Потому джсон содержит наборы простых узлов: *расходник+саморез+условие+формула*. Фронт просеивает по условию расходники и после этого сам группирует их (чтобы вариации одинакового расходника и вариации саморезы склеились для пользователя в попап выбора).

Джсон основной структуры (ключ *suits*) представляет собой дерево с группировкой по поверхности:

```
{
  "1239": [ // Код поверхности (дерево, газобетон, стена+к, ...)
    {
      "rashodnik": 1, // код на "расходник" с лингвистикой и настройками
      "rashodnikVariation": 1, // вариация расходника (фактически, товар)
      "samorez": 1, // код на "саморез" с лингвистикой
      "samorezVariation": 5,
      "formula": 1,
      "rules" : 1
    },
    {
      "rashodnik": 1, // код на "расходник" с лингвистикой и настройками
      "rashodnikVariation": 1, // вариация расходника (фактически, товар)
      "samorez": 1, // код на "саморез" с лингвистикой
      "samorezVariation": 6,
      "formula": 1,
      "rules" : 1
    },
  ]
}
```

Правила, условия (в джсоне: rules)

Не все комплектующие, описанные для определённой сущности (стена, потолок) и поверхности (газобетон, камень, ...) попадают в обрешётку. Каждая комплектующая сопровождается правилом, по которому эта комплектующая попадает в обрешётку.

Правило представляет собой одно условие или комбинацию условий. В качестве условия в комбинации может быть другая комбинация. Правила подвязаны к определённому свойству товара (для ширины доски, для толщины доски, прифиля, материала,...). Список правил для параметров формируют сет, которых может быть несколько. Чтобы комплектующая попала в обрешётку, должен сработать **любой (хотя бы "один из...")** сетов.

На скриншоте ниже набор сетов.

Первый сет говорит о том, что комплектующая попадёт в обрешётку, если:

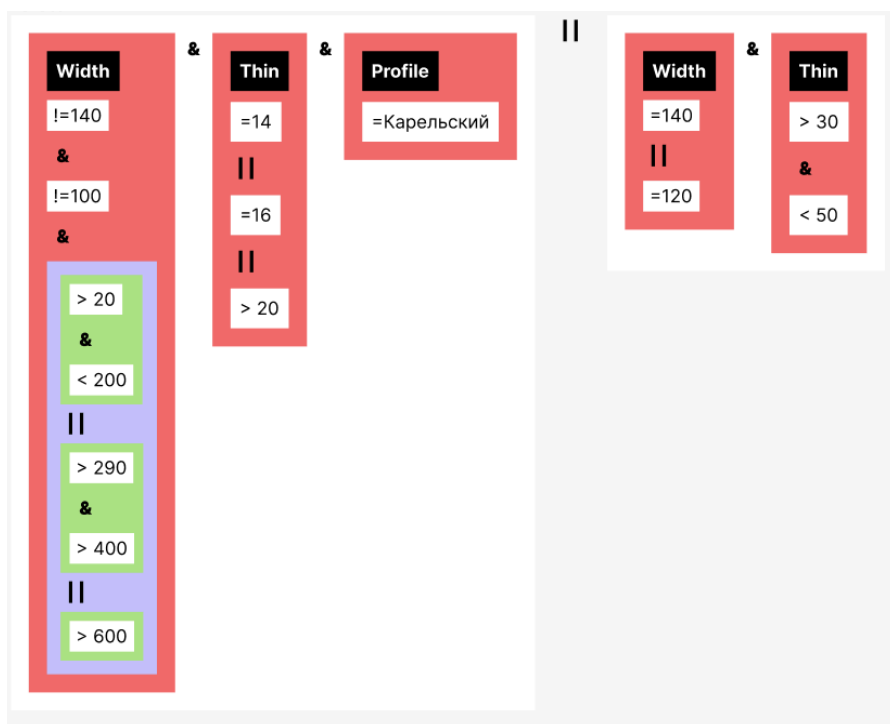
Ширина доски $\neq 140$ и $\neq 100$ и (находится между 20 и 200 или между 290 и 400 или больше 600)

- + толщина доски = 14 или 16 или больше 20
- + профиль доски "Карельский"

Второй сет говорит о том, что комплектующая попадёт в обрешётку, если:

Ширина доски = 140 или 120

- + толщина доски между 30 и 50



Уровень 1 (комбинация сетов) — это **всегда "ИЛИ"** (или один сет или другой или третий, ...)

Уровень 2 (разные параметры, каждый — в зависимости от: ширины доски, толщины доски, профиля, ...) — это **всегда "И"** (из примера первого сета: должны выполняться условия **И** для ширины **И** для толщины **И** для профиля).

В джсоне:

```
[
  {
    > "work_width": {"logic": "AND"...},
    // AND
    > "profile": {"logic": "OR"...},
    // AND
    > "thin": {"logic": "OR"...}
  },
  // OR
  {
    > "work_width": {"COND": "!"...},
    // AND
    > "thin": {"logic": "AND"...}
  }
]
```

Особенности форматирования:

Понимание необходимо для разбора и анализа джсона с правилами.

Простое условие

```
"work_width": {
  "COND": "!",
  "VALUE": 140
},
```



Простое условие — это объект с ключами **COND** и **VALUE**.

Как мы видим на скриншоте выше, в случае, когда для параметра (width) нужно указать только одно условие, джсон будет содержать объект с простым условием.

Набор условий

```
"work_width": {
  "logic": "AND",
  "rules": [
    > {"COND": "!"...},
    > {"COND": "!"...},
    > {"logic": "OR"...}
  ]
},
```



Набор условий — объект с ключами **logic** и **rules**.

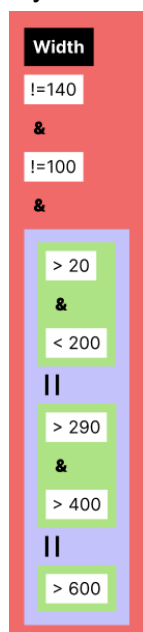
- **logic** указывает, какая логика действует для правил на данном уровне
- **rules** содержит набор условий (где, напоминая, любое из условий может в свою очередь быть как простым условием так и набором)

На скриншоте выше мы видим, что текущий уровень условий содержит два простых условия (распознаём по ключу **COND**) и набор условий (присутствует ключ **logic**).

Разбор условий (подходит/нет) скорее всего нужно будет делать через рекурсию.

!!! Для проверки условий нужны значения общих переменных. Описано ниже, в подразделе Общие переменные

На этапе проектирования пришли к выводу, что такое ветвление является максимально возможно глубоким:



Пример джсона: <https://pastebin.com/XCqDBQez>

Формулы

Формулы нужны для того, чтобы определить, сколько единиц расходника понадобится для покрытия обрешёткой заданной площади.

Есть несколько уровней переменных.

Общие переменные

Общие переменные задаются прямо в коде компонента и вычисляются самими первыми (может быть даже в какой-то компьютер-объект).

!!! Важно. Общие переменные используются не только в формулах но и в условиях, а значит должны вычисляться где-то на более ранней стадии

CPS - количество копий сущности (3 стены, 4 фальшбалки, ...)

S - площадь или рабочая поверхность

L - рабочая длина (длина стены, потолка, ...)

H - рабочая высота

productId = this.productId // код выбранного материала (конкретная вагонка, конкретный планкен, ...). Используется для получения переменной *product*.

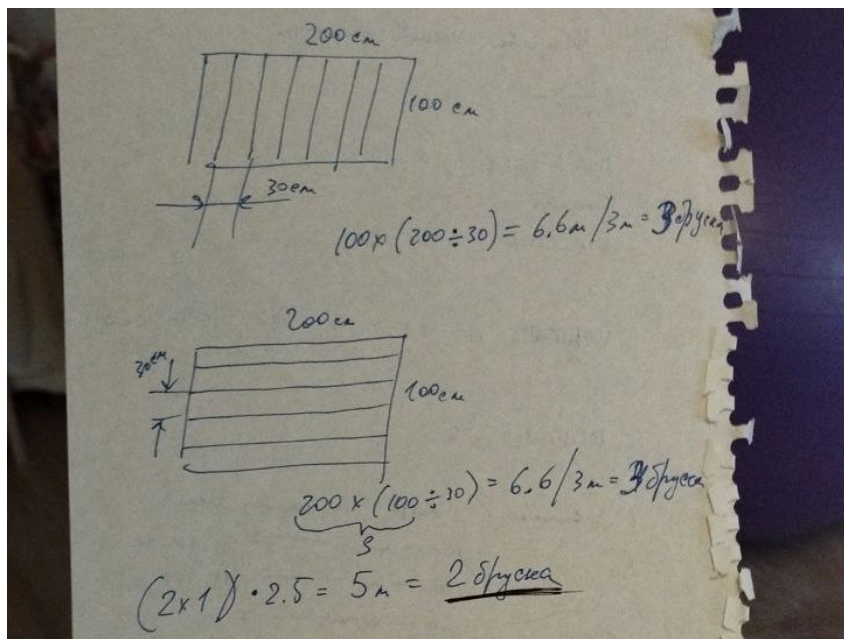
product = this.db.products[productId] // ссылка на продукт по его коду. Используется для получения переменных ниже.

woodWidth = parseInt(product["WOOD_WIDTH_VALUE"]) // вычисление рабочей ширины доски

woodThickness = parseInt(product["WOOD_THICKNESS_VALUE"]) // вычисление толщины доски

Пользовательские значения

Параметр, который пользователь может менять на странице. Есть как минимум один такой параметр — **U_OSTP** — шаг обрешётки (расстояние между брусками при креплении на поверхности)



Динамические переменные

По-факту, мини-формулы. Например:

D_OBRPM = Math.ceil(S*(100/U_OSTP)) - сколько погонных метров обрешётки нужно чтобы покрыть установленную площадь, зависимость от шага обрешётки (U_OSTP)

Формулы

Самый верхний (последний) уровень в вычислениях. Когда собраны общие переменные, собраны пользовательские значения и вычислены динамические переменные, срабатывает общая формула комплектующей, которая и даёт ответ на вопрос "сколько единиц расходника нужно".

Например, формула кляймеров: **D_OBRPM * 5** — 5 штук на погонный метр.

Чтобы её вычислить (а вычисление представляет собой банальный `eval(calculateFormula)`), должна быть вычислена переменная **D_OBRPM** (сколько погонных метров обрешётки нужно для указанной площади).

Чтобы вычислить **D_OBRPM**, нужны:

- общая переменная **S**
- пользовательское значение **U_OSTP**

Чтобы вычислить формулу, нужно заменить все сложные части (с префиксом **D_**) в ней на простые и выполнить `eval`.

То есть формулу

D_OBRPM * 5

...мы пересобираем как

(Math.ceil(S*(100/U_OSTP))) * 5

...и выполняем *eval*

Полученное значение:

- трансформируем согласно настройкам продажи
 - если товар продаётся **штуками** — ура, мы уже получили значение;
 - если товар продаётся **упаковками**, нужно определить количество упаковок (на выходе получаем два значения: в штуках и в упаковках);
 - если **погонными метрами**, нужно вычислить количество штук. Например, мы определили, что нужно 20 метров обрешётки, а бруски, которые входят в обрешётку, — трёхметровые. Значит нужно 7 штук брусков (на выходе получаем два значения: в штуках и в метрах погонных);
- дополняем лингвистикой (из настроек расходника)
 - если товар продаётся штуками — "X шт";
 - если товар продаётся упаковками — "Y **упаковок** (X шт)";
 - если погонными метрами — "Y **метров обрешётки** (X шт)".

Админка обрешёток

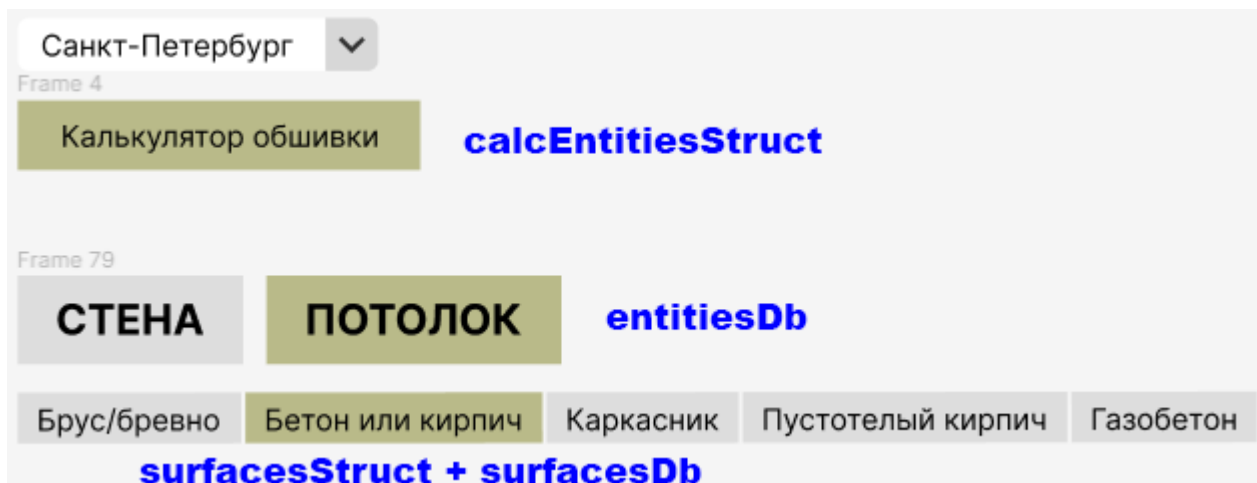
Оболочка для админки находится тут:

https://dev.lesobirzha.ru/bitrix/admin/obreshetka_admin.php

Дизайн:

<https://www.figma.com/file/rBWrl0q6V4Med2ZEfuMyvS/%D0%9B%D0%9A%D0%9C-%D0%9A%D0%B0%D0%BB%D1%8C%D0%BA%D1%83%D0%BB%D1%8F%D1%82%D0%BE%D1%80%D0%BE%D0%B2-%D0%90%D0%B4%D0%BC%D0%B8%D0%BD%D0%BA%D0%B0?type=design&node-id=789%3A349&mode=design&t=DE6AL6NSZoGExN4I-1>

Поверхности для крепления обрешётки



Ключи:

структура сущностей calcEntitiesStruct по кодам калькуляторов:

```
▼ calcEntitiesStruct: {calc_obshivka:
  ▼ calc_obshivka: ["1239", "1240"]
    0: "1239"
    1: "1240"
  ▼ calc_obshivka_spb: ["1298"]
    0: "1298"
}
```

+ **БД сущностей entitiesDb** — расшифровка сущностей по айди (стена, потолок)

```
▼ entitiesDb: {1239: {CODE: "wall", NAME: "Ст"},
  ► 1239: {CODE: "wall", NAME: "Стена"}
  ► 1240: {CODE: "ceiling", NAME: "Потолок"}
  ► 1298: {CODE: "wall", NAME: "Стена"}
```

+ **структура поверхностей surfacesStruct** — поверхности крепления обрешётки (Бревно, Газобетон, ...) относительно сущностей (Стена, Потолок, ...). На текущий момент даны ключи как по символному коду (удобно для публички) так и по числовому коду (для админки)

```
▼ surfacesStruct: {,...}
  ▼ calc_obshivka: {1239: [1343, 1336, 1337, 1341, 1342], 1
    ► 1239: [1343, 1336, 1337, 1341, 1342]
    ► 1240: [1340, 1339, 1344, 1345]
    ► ceiling: [1340, 1339, 1344, 1345]
    ► wall: [1343, 1336, 1337, 1341, 1342]
    ► calc_obshivka_spb: {1298: [1347, 1348, 1349, 1350, 1351]}
```

+ **БД поверхностей surfacesDb** *.


```

▼ surfacesDb: {1336: {NAME: "Обрешетка для каркасника", CODE: "karkas", :
  ▼ 1336: {NAME: "Обрешетка для каркасника", CODE: "karkas", SHORT_NAME:
    CODE: "karkas"
    NAME: "Обрешетка для каркасника"
    SHORT_NAME: "Каркас"
  ▶ 1337: {NAME: "Обрешётка для бетона или полнотелого кирпича", CODE: ":
  ▶ 1339: {NAME: "Обрешётка для дерева с коммуникациями", CODE: "wood_co
  ▶ 1340: {NAME: "Обрешётка для дерева", CODE: "wood", SHORT_NAME: "Дере
  ▶ 1341: {NAME: "Обрешётка для пустотелого кирпича", CODE: "brick", SHO
  ▶ 1342: {NAME: "Обрешётка для газобетона", CODE: "concrete", SHORT_NAMI

```

*) SHORT_NAME используется для рендера кнопок

Брус/бревно Бетон или кирпич Каркасник Пустотелый кирпич Газобетон

Панель с расходниками

Обрешетка Дюбеля × Уголок в бетон × Кляймеры × +

БД расходников (для кнопочки "+") — в ключе **consumables**

```

▼ consumables: {1: {ID: "1", UF_NAME: "Уго
  ▶ 1: {ID: "1", UF_NAME: "Уголок в бетон".
  ▶ 2: {ID: "2", UF_NAME: "Обрешётка", UF_
  ▶ 3: {ID: "3", UF_NAME: "Кляймер", UF_AC

```

Расходник, который отмечен как "Обрешётка", публикуется с флажком:

```

▶ 1: {ID: "1", UF_NAME: "Уголок в б
▼ 2: {ID: "2", UF_NAME: "Обрешётка"
  ID: "2"
  SELL_BY_VALUE: "length"
  UF_ACTIVE: "1"
  UF_BASE: "1"
  UF_LANG_CASE_1: ""
  UF_LANG_CASE_2: ""

```

Такой расходник нельзя удалить, он основа всей обрешётки.

Список уже размещённых на поверхности расходников строится динамически по списку из ключа **suits**:

```

▼ suits: {, ...}
  ▼ 1336: [{rashodnik: "1", rashodnikVariation: "1", samorez: '
    ▶ 0: {rashodnik: "1", rashodnikVariation: "1", samorez: '
    ▶ 1: {rashodnik: "2", rashodnikVariation: "3", samorez: '
  ▼ surfacesDb: {1336: {NAME: "Обрешетка для каркасника", CODE
    ▼ 1336: {NAME: "Обрешетка для каркасника", CODE: "karkas",
      CODE: "karkas"
      NAME: "Обрешетка для каркасника"
      SHORT_NAME: "Каркас"

```

Расшифровка скрина выше:

На поверхности 1336 (Каркас|Обрешётка для каркасника) размещено два условия. В одном условии расходник №1, во втором расходник №2.

Расходник №1 по базе из ключа consumables — Уголок в бетон.

Расходник №2 по базе из ключа consumables — Обрешётка.

Один и тот же расходник может встречаться в списке сколько угодно раз, конкретно здесь (при построении панели) нас интересует только то, какой расходник используется (его ID), всё-равно сколько раз.

Таким образом, на панели будет только два расходника:

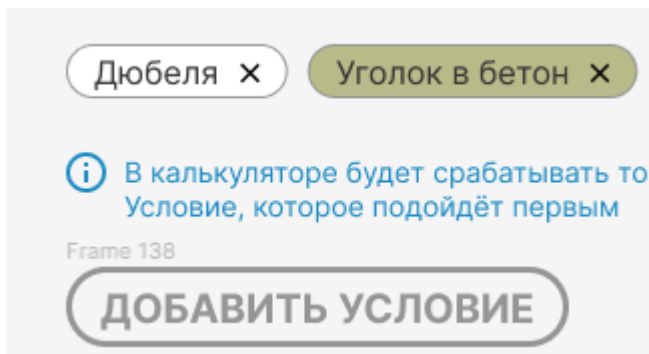


...и кнопка "+".

Кнопка "+" пока ещё отображается, поскольку в ключе consumables, где на момент, который рассматривается в примере, три расходника и доступен (не размещён на панельке) ещё "Кляймер". Когда кляймер будет размещён на странице, кнопка "+" перестанет отображаться.

Условия расходника

Когда мы добавляем расходник через кнопку "+", нам отображается напоминание (видно всегда) и кнопка "добавить условие"



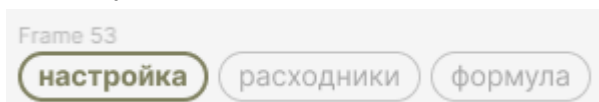
Управление вариациями расходника (и саморезами) происходит через настройки разных **Условий**.

Для понимания, пример двух условий расходника *Уголок в бетон*:

1. Уголок 40*30 добавляется в обрешётку для **планкена** с толщиной доски **больше 40 мм**
2. Для всех остальных случаев в обрешётку добавляется уголок 30*30

Условия тут самая главная организационная сущность.

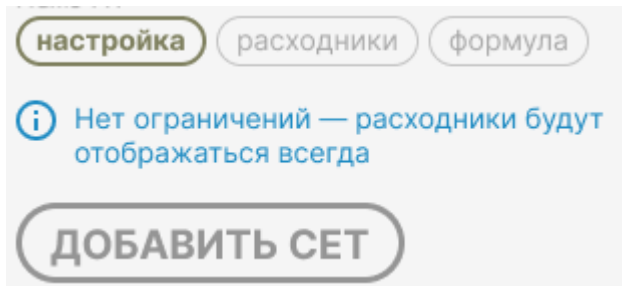
Каждое условие имеет мини-панель



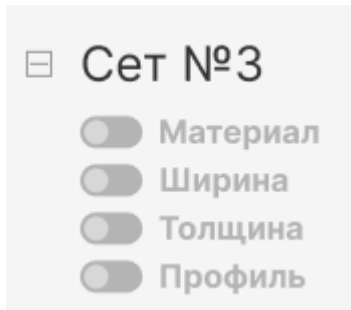
Настройка

Условие, в котором нет ни одного сета, срабатывает для всех вариантов (ширины доски, толщины, профиля, материала,...).

В пустом (или новом) условии мы видим информационный блок и кнопку "Добавить сет"



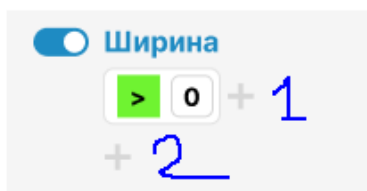
В новом сете сразу отображаются все доступные параметры



Параметры берутся из джсона **rulesKeys**

```
▼ rulesKeys: {,...}
  ▼ material: {ID: "3", UF_NAME: "Материал", UF_CODE: "matheri
    ID: "3"
    TYPE_CODE: "list"
    UF_CODE: "material"
    UF_CUSTOM_ORIGIN: "catalog:material"
    UF_ENUM_ORIGIN: ""
    UF_NAME: "Материал"
    UF_RELATIVE_CHAIN: "product:IBLOCK_SECTION_CODE"
  ▼ wood_thickness: {ID: "2", UF_NAME: "Толщина доски", UF_CODE
    ID: "2"
    TYPE_CODE: "number"
    UF_CODE: "wood_thickness"
    UF_CUSTOM_ORIGIN: ""
    UF_ENUM_ORIGIN: ""
    UF_NAME: "Толщина доски"
    UF_RELATIVE_CHAIN: "D:woodThickness"
  ► work_width: {ID: "1", UF_NAME: "Ширина доски", UF_CODE: "wc
```

Каждый параметр при активации представляет собой мини-конструктор



Блок на белом фоне — **условное звено** — это панелька с:

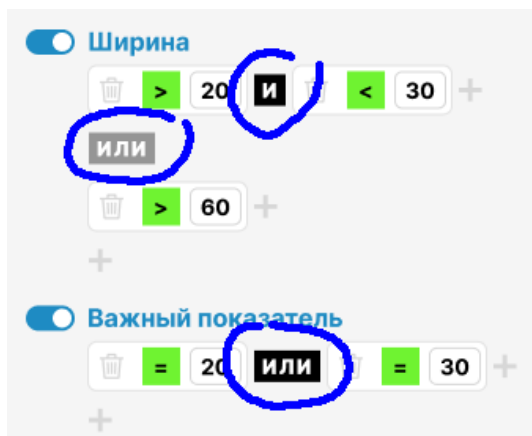
- выбором оценки (<, >, <=, >=, =, не)
- выбором значения. Значение — это либо **поле для ввода числа** (как на скриншоте выше), либо **"выпадающий список"**, замаскированный в слово:



На скриншоте с формулой: первый плюс (горизонтальное расширение) добавляет кнопку логики и ещё одно **условное звено**. В конце цепочки всегда отображается плюс, чтобы добавить можно было новое условное звено и продолжить цепочку.

Второй плюс (вертикальное расширение) добавляет слой логики, разделённый с предыдущим кнопкой логики.

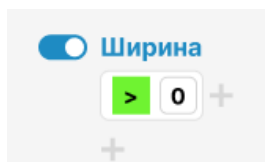
Кнопки логики:



...принимают значения:

- И (по-умолчанию)
- ИЛИ

Самое первое звено удалить нельзя



Тип значения (число или список) для построения **условного звена** берётся из ключа TYPE_CODE каждого из ключей

```
▼ rulesKeys: {work_width: {ID: "1", UF_NAME: "Ширина доски", UF_CODE: "work_width"},
  ▼ matherial: {ID: "3", UF_NAME: "Материал", UF_CODE: "matherial", ID: "3"}
    ● TYPE_CODE: "list"
    UF_CATALOG_ENUM_ORIGIN: null
    UF_CODE: "matherial"
    UF_CUSTOM_ORIGIN: "catalog:matherial"
    UF_ENUM_ORIGIN: ""
    UF_NAME: "Материал"
    UF_RELATIVE_CHAIN: "product:IBLOCK_SECTION_CODE"
  ► wood_profile: {ID: "4", UF_NAME: "Профиль", UF_CODE: "wood_profile"}
  ▼ wood_thickness: {ID: "2", UF_NAME: "Толщина доски", UF_CODE: "wood_thickness", ID: "2"}
    ● TYPE_CODE: "number"
    UF_CATALOG_ENUM_ORIGIN: null
    UF_CODE: "wood_thickness"
    UF_CUSTOM_ORIGIN: ""
    UF_ENUM_ORIGIN: ""
    UF_NAME: "Толщина доски"
    UF_RELATIVE_CHAIN: "woodThickness"
  ► work_width: {ID: "1", UF_NAME: "Ширина доски", UF_CODE: "work_width"}}
```

Для ключей типа список (list) отдельно предусмотрен джсон-ключ **rulesVariants** со списком значений на выбор, сгруппированных по тому же ключу, что в **rulesKeys**:

Расходники

В условии менеджер видит все вариации расходника (а также все вариации саморезов, которые прикреплены к расходникам (если прикреплены)):

настройка **расходники** формула

☒ Крепежный уголок
равносторонний, широкой KUR
60×500

☐ Крепежный уголок
равносторонний, широкой KUR
60×120

☐ саморезы

☒ Крепежный уголок
равносторонний, широкой KUR
60×500

☐ саморезы

☒ Саморезы KREG 3,5x50,8 мм
для террасной доски и
наличников (2 штуки на уголок)

☐ Саморез оцинкованный Bit-PZ2
4x20 (2 штуки на уголок)

☒ Саморез оцинкованный Bit-PZ2
4x40 (2 штуки на уголок)

Базы данных для получения данных по расходникам и вариациям:

- consumables

```
▼ consumables: {1: {ID: "1", UF_NAME: "Уголок в бетон", L
  ► 1: {ID: "1", UF_NAME: "Уголок в бетон", UF_ACTIVE: "1"
  ► 2: {ID: "2", UF_NAME: "Обрешётка", UF_ACTIVE: "1", UF
  ► 3: {ID: "3", UF_NAME: "Кляймер", UF_ACTIVE: "1", UF_S
```

...расходники, по сути — [логические группы](#), которые содержат настройки

```
▼ 1: {ID: "1", UF_NAME: "Уголо
  ID: "1"
  SELL_BY_VALUE: "sht"
  UF_ACTIVE: "1"
  UF_LANG_CASE_1: "уголок"
  UF_LANG_CASE_2: "уголка"
  UF_LANG_CASE_5: "уголков"
  UF_LANG_COMMON: ""
  UF_NAME: "Уголок в бетон"
  UF_SORT: "10"
```

...из которых в админке нужны: название и сортировка

- consumVariationsDb

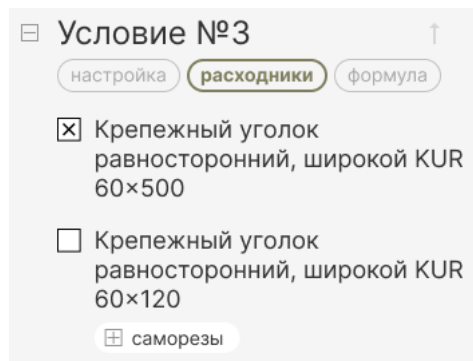
```
▼ consumVariationsDb: {1: {ID: '
  ▼ 1: {ID: "1", UF_ACTIVE: "1"
    ID: "1"
    PRODUCT_ID: "6572"
    UF_ACTIVE: "1"
    UF_CONSUMABLE_ID: "1"
    UF_SORT: "10"
  ► 2: {ID: "2", UF_ACTIVE: "1"
  ► 3: {ID: "3", UF_ACTIVE: "1"
  ► 4: {ID: "4", UF_ACTIVE: "1"
```

...из которых в админке нужны: код расходника, сортировка и код товара из каталога

- **consumVariationsProducts** — база данных товаров из каталога

```
▼ consumVariationsProducts: {396: {NAME: "Брусok строганный из  
▼ 396: {NAME: "Брусok строганный из сосны 40x20x2000 прямоугo.  
DETAIL_PAGE_URL: "/katalog/brus/brusok-strogannyi-iz-sosny  
LENGTH: 2  
NAME: "Брусok строганный из сосны 40x20x2000 прямоугольнoгo  
PACK_PIECES: 1  
► PRICES: {1: "58.00", 2: "47.00"}  
► 5062: {NAME: "Кляймер усиленный №6", DETAIL_PAGE_URL: "/kat  
► 6572: {NAME: "Крепежный уголок с ребром жесткости КУ 50x35".  
► 10766: {NAME: "Перила для террасных ограждений из лиственни
```

Именно название товара публикуется в группе расходника на месте вариации



- **samorezi**

samorezVariationsDb

samorezVariationsProducts

```
► samorezVariationsDb: {,...}  
► samorezVariationsProducts: {25441: {NAME: "Саморез оцинкованный Bi  
► samorezi: {1: {ID: "1", UF_LANG_FOR: "уголок", UF_ACTIVE: "1", CON:
```

...по своей сути аналогичные описанным выше ключам для расходников.

Напомню, что расходники заезжают в джсоне в линейном (а не древовидном дереве) и задача по построению дерева ложится на фронт. Описано [в концепции выше](#).

Менеджер выбирает, какие уголки и саморезы остаются в данном условии (отмечает чекбоксами), в этом заключается вся настройка расходников Условия.

Для построения подписи ...(2 штуки на уголок) берутся поля из настроек самореза:

☒ Крепежный уголок
равносторонний, широкой KUR
60×500

саморезы

☒ Саморезы KREG 3,5x50,8 мм
для террасной доски и
наличника (2 штуки на уголок)

☐ Саморез оцинкованный Bit-PZ2
4x20 (2 штуки на уголок)

☒ Саморез оцинкованный Bit-PZ2
4x40 (2 штуки на уголок)

▼ samorezi: {1: {ID: "1", UF_LANG_FOR: "уголок", UF_ACTIVE: "1",
UF_ID: "1", UF_LANG_FOR: "уголок", UF_ACTIVE: "1",
CONSUM_VARIANT_ID: "1",
ID: "1",
UF_ACTIVE: "1",
UF_LANG_FOR: "уголок",
UF_QUANTITY: "2"}, 2: {ID: "2", UF_LANG_FOR: "обрешетка", UF_ACTIVE: "1",
UF_ID: "2", UF_LANG_FOR: "обрешетка", UF_ACTIVE: "1",
CONSUM_VARIANT_ID: "2",
ID: "2",
UF_ACTIVE: "1",
UF_LANG_FOR: "обрешетка",
UF_QUANTITY: "1"}}

Саморезы в паре

Очень редкая ситуация, но она актуальна и в текущей (старой) версии компонента обрешётки уже реализована:

Материал обшивки

Материал

Порода

Сорт

Длина

Планкен

Лиственница

С/Д

3,00

Планкен из лиственницы скошенный сорт CD 140x20x3000

Запас на порезку 10%

Планкен из лиственницы скошенн... — понадобится 48 упаковок, 288 шт.

Бюджет: 98 208 рублей

Обрешетка

Обрешётка для бетонн или полнотелого кирпича

500 уголков
[Крепежный уголок с ребром жесткости KU 50x35](#)

500 саморезов в под дюбели в бетон
[Саморез оцинкованный Bit-PZ2 4,5x50](#)

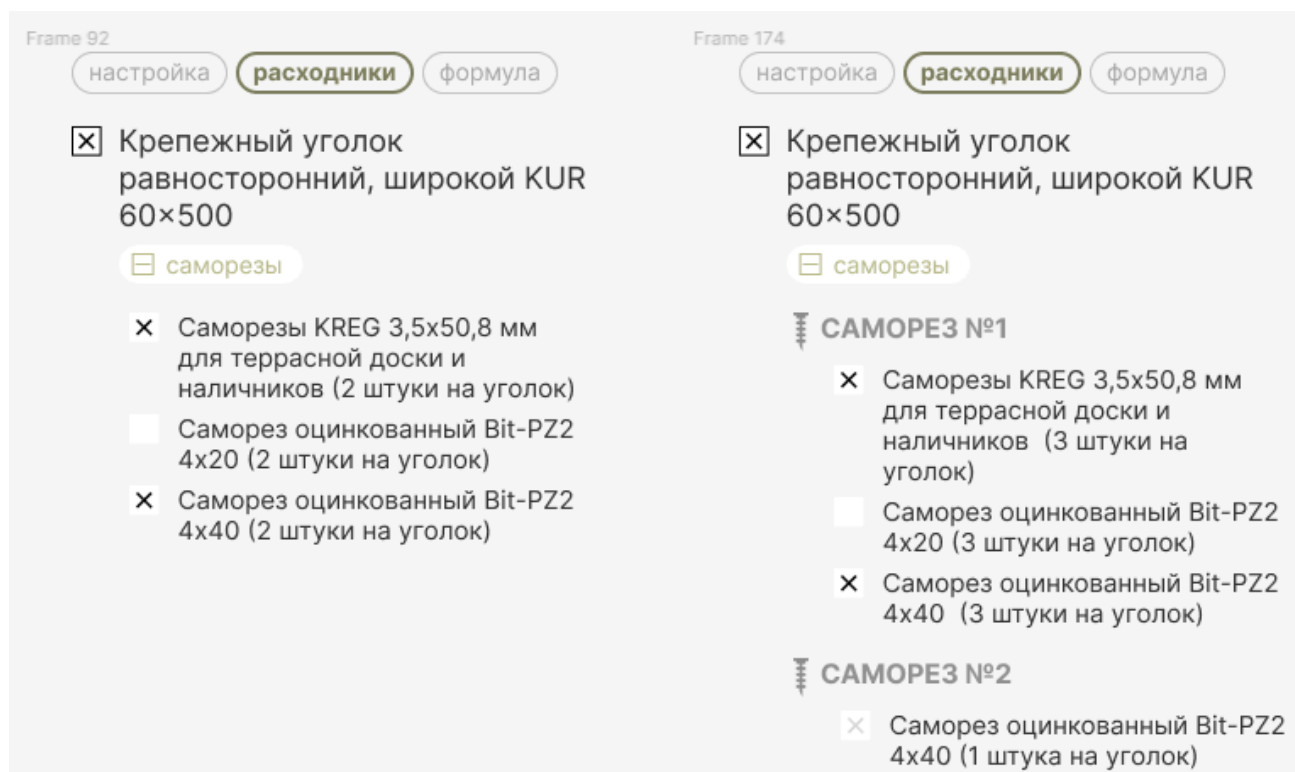
1400 крепежей
[Усиленный крепеж для планкена и террас LesFix 190x18x1,0 со стопором \(зазор с](#)

9800 саморезов под крепежи
[Саморез оцинкованный Bit-PZ2 4x20](#)
[Саморез оцинкованный Bit-PZ2 4x40](#)

Бюджет: 85 902 рублей.

Конструкция крепежа такова, что в него идёт не один тип самореза, а два.

Визуальное различие для менеджера:



Слева типичная ситуация, встречающаяся в большинстве случаев: у вариации расходника есть саморез и менеджер видит список вариации самореза.

Справа саморезы представлены в виде подгрупп с нумерацией: "Саморез №1" и "Саморез №2". Каждый из них **должен быть представлен в Условии**. То есть правило обязательности "минимум одна вариация самореза должна присутствовать" работает уже в рамках каждой подгруппы. Не должно быть так, что выбрана вариация самореза для подгруппы "Саморез №1" и не выбрано ни одной вариации для "Саморез №2".

Визуальное различие в джсоне:

В общей ситуации под одну вариацию расходника относится один саморез (с его вариациями). Если же под одну вариацию расходника в джсоне отмечено несколько саморезов (не вариаций саморезов а именно саморезов), это как раз случай с "саморезами в паре".

Пример обычной ситуации:

```
▼ samorezi: {1: {ID: "1", UF_LANG_FOR: "уголок", UF_ACTIVE: "1", CONSUM_VARIATION_ID: "1"}, 2: {ID: "2", UF_LANG_FOR: "обрешетку", UF_ACTIVE: "1", CONSUM_VARIATION_ID: "3"}, 3: {ID: "3", UF_LANG_FOR: "уголок", UF_ACTIVE: "1", CONSUM_VARIATION_ID: "1"}}, suits: {...}}
```

...один саморез под вариацию расходника с айди 3

Пример ситуации "саморезов в паре":

```
▼ samorezi: {1: {ID: "1", UF_LANG_FOR: "уголок", UF_ACTIVE: "1", CONSUM_VARIATION_ID: "1"}, 2: {ID: "2", UF_LANG_FOR: "обрешетку", UF_ACTIVE: "1", CONSUM_VARIATION_ID: "3"}, 3: {ID: "3", UF_LANG_FOR: "уголок", UF_ACTIVE: "1", CONSUM_VARIATION_ID: "1"}}, suits: {...}}
```

...под одну и ту же вариацию расходников — два разных самореза

Источник данных

Сохранённые данные приходят в джсон-ключе **suits**.

Это массивы записей вариаций расходников, распределённые по поверхностям

Скриншот с небольшими пояснениями: (увеличенная версия <https://snipboard.io/IPrTsO.jpg>)

The screenshot displays a JSON structure for 'suits' and a corresponding UI for material selection. Blue arrows indicate the mapping between the JSON data and the UI elements:

- surfacesDb** (JSON) maps to the **СТЕНА** and **ПОТОЛОК** tabs in the UI.
- formula** (JSON) maps to the **Уголок в бетон** button in the UI.
- consumables** (JSON) maps to the **Уголок в бетон** button in the UI.
- rules** (JSON) maps to the **Уголок в бетон** button in the UI.

Каждая запись в массиве — одна вариация расходника + одна вариация самореза (если такой предусмотрен).

Ситуация типа такой:

The screenshot shows a configuration window for 'Условие №3' (Condition №3). The 'расходники' (Materials) tab is active, displaying a list of materials and fasteners:

- Крепежный уголок** (Fastening angle)
 - равносторонний, широкой KUR 60×500
- саморезы** (Screws)
 - САМОРЕЗ №1** (Screw №1)
 - Саморезы KREG 3,5x50,8 мм для террасной доски и наличников
 - Саморез оцинкованный Bit-PZ2 4x20
 - Саморез оцинкованный Bit-PZ2 4x40
 - САМОРЕЗ №2** (Screw №2)
 - Саморез оцинкованный Bit-PZ2 4x40

...будет представлена тремя записями (а не одним и не четырьмя (хотя мы видим четыре чекбокса)):

вариация уголка + вариация самореза 1

вариация уголка + вариация самореза 2

вариация уголка + вариация самореза 3

...да, с дублями записей правил и условий.

Если вариация расходника не предполагает саморезы, вариации саморезов будут пустыми в записях.

Система не должна допускать ситуацию, когда не выбрана ни одна вариация расходника. Если вариация расходника предполагает саморезы, система не должна допускать ситуацию, когда ни одна вариация самореза не выбрана.

Пример того, как система следит за обязательностью выбора:

Frame 92

настройка **расходники** формула

☒ ☐ Крепежный уголок
равносторонний, широкой KUR
60×500

☐ Крепежный уголок
равносторонний, широкой KUR
60×500

☐ саморезы

САМОРЕЗ №1

☐ Саморезы KREG 3,5x50,8 мм
для террасной доски и
наличников

☐ Саморез оцинкованный Bit-PZ2
4x20

☒ ☐ Саморез оцинкованный Bit-PZ2
4x40

САМОРЕЗ №2

☐ Саморез оцинкованный Bit-PZ2
4x40

...серые чекбоксы снять нельзя. Какой чекбокс оказывается после действий менеджера последним, тот и дизейблится.

Формулы

Формулы заезжают (и накапливаются и отправляются на сохранение) в джсон-ключе **formulas**:

```
▼ formulas: {1: {ID: "1", UF_FORMULA: "S*5"}, 2: {ID: '
  ► 1: {ID: "1", UF_FORMULA: "S*5"}
  ► 2: {ID: "2", UF_FORMULA: "Math.ceil(H*2*CPS)"}
```

Формула, собранная для каждого отдельного Условия добавляется в джсон отдельной записью.

Даже если уже есть точно такая же формула в джсоне.

То есть после формирования такого набора Условий:

Дюбеля x Уголок в бетон x Обрешетка

В калькуляторе будет срабатывать то Условие, которое подойдёт первым

Frame 138

Условие №1
Отображается всегда

Условие №2
Ширина: >20 и <30, Толщина: 15, 18;
Профиль: Сибирский, Карельский
или
Ширина: 120, 140, >400

Условие №3
настройка **расходники** формула

☒ Крепежный уголок
равносторонний, широкой KUR
60×500

☐ Крепежный уголок
равносторонний, широкой KUR
60×120

...ключ formulas увеличится на три записи. Даже если формулы у всех условий одинаковы.

Если в публичной части с формулами всё понятно (просто [подставили данные и выполнили eval](#)), в админке всё сложнее — тут предусмотрен мини-конструктор.

Формула:

*0.350+4

Переменные:

площадь, м2 м.п. обрешетки длина стены, м высота стены, м

ширина доски, мм толщина доски, мм

Формула представляет собой цепочку полей ввода (ПВ) и звеньев-переменных (ЗП). Это всегда шахматка, которая и начинается и заканчивается полем ввода:

ПВ + ЗП + ПВ

Добавление в формулу новой переменной всегда увеличивает цепочку парой ЗП + ПВ:

ПВ + ЗП + ПВ + ЗП + ПВ

ПВ + ЗП + ПВ + ЗП + ПВ + ЗП + ПВ

(...)

Звено-переменная является списком выбора (выпадающий список с перечнем переменных).

Удаление звена-переменной приводит к тому, что содержимое текстовых полей, которые его окружают, склеивается в одно поле.

!!! Нельзя удалить звено-переменную, если оно осталось одно в формуле.

Собранная формула к текстовому виду представляет собой готовую строку с js-кодом.

Удаление Условия удаляет связанные записи, в частности запись из **formulas** (потому что на эту запись больше никто "не смотрит").

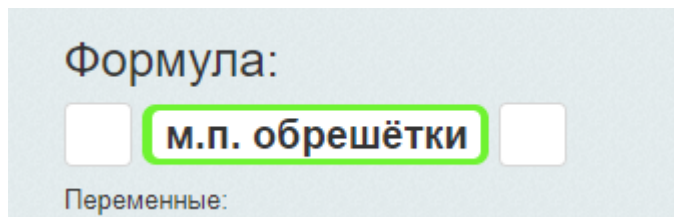
Добавление

При добавлении новой формулы:

- *ключ объекта* равняется **самому большому ключу в объекте + 1**;
- ключ ID внутри объекта равняется *ключу объекта*.
- заполняется формула по-умолчанию, берётся из ключа formulaAdminDefault

```
► entitiesDb: {1239: {CODE: "wall", I
  formulaAdminDefault: "OBRPM*4"
  formulaDynamics: {,...}}
```

Визуально выглядит так:



Переменные

Список переменных составляется из:

- Общих переменных - ключ **formulasCommonMembers**

```
▼ formulasCommonMembers: {CPS: {ID: "1", UF_CODE: "CP
  ► CPS: {ID: "1", UF_CODE: "CPS", UF_SORT: "10", UF_
  ▼ H: {ID: "4", UF_CODE: "H", UF_SORT: "40", UF_FORM
    ID: "4"
    UF_CODE: "H"
    UF_FORMULA: "wallHeight"
    UF_NAME: "Высота стены"
    UF_SORT: "40"
    UF_USE_IN_ADMIN_CONSTRUCTOR: "1"
  ► L: {ID: "3", UF_CODE: "L", UF_SORT: "30", UF_FORM
  ▼ S: {ID: "2", UF_CODE: "S", UF_SORT: "20", UF_FORM
    ID: "2"
    UF_CODE: "S"
    UF_FORMULA: "this.wallSquare"
    UF_NAME: "Площадь"
    UF_SORT: "20"
    UF_USE_IN_ADMIN_CONSTRUCTOR: "1"
```

...используются только те записи, у которых UF_USE_IN_ADMIN_CONSTRUCTOR равняется 1

- Динамических переменных - ключ **formulaDynamics**

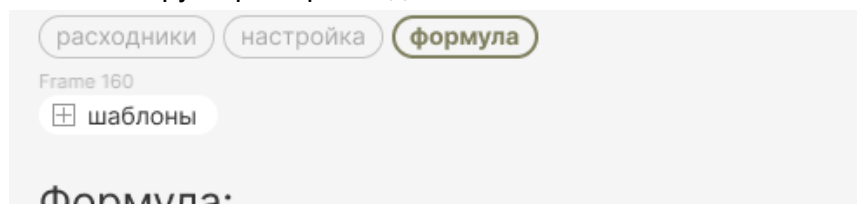
```
▼ formulaDynamics: {,...}
  ▼ OBRPM: {ID: "1", UF_CODE: "OBRPM", UF_FORM
    ID: "1"
    UF_CODE: "OBRPM"
    UF_FORMULA: "Math.ceil(S*(100/U_OSTP))"
    UF_NAME: "м.п. обрешётки "
```

...!!! которые в текстовом скрине формулы идут с префиксом **D_**.

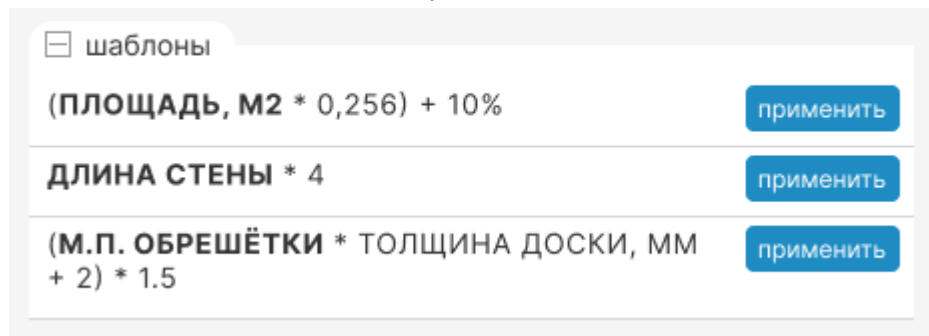
Например, переменная OBRPM в формуле "м.п. обрешётки * 4.5" будет выглядеть так: **"D_OBRPM*4.5"**.

Шаблоны

Мини-конструктор сопровождается блоком "шаблоны"



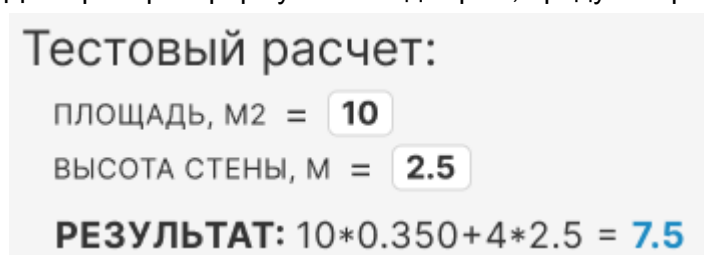
В нём отображаются все формулы из ключа formulas, которые внесены туда на текущий момент:



Менеджер может выбрать шаблон и с помощью кнопки "применить" заменить текущую формулу на ту, которую выбрал.

Результаты вычисления

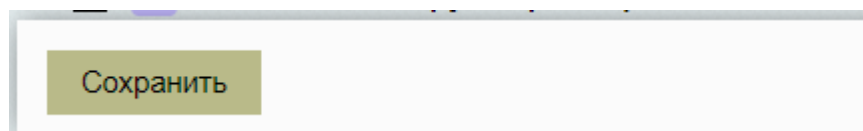
Для проверки формулы менеджером, предусмотрен специальный блок



...куда менеджер может внести значения и свериться со своими вычислениями, убедившись, что формула составлена верно.

Сохранение

По аналогии с админой ЛКМ, в нижней части экрана расположена панель с кнопкой "Сохранить"

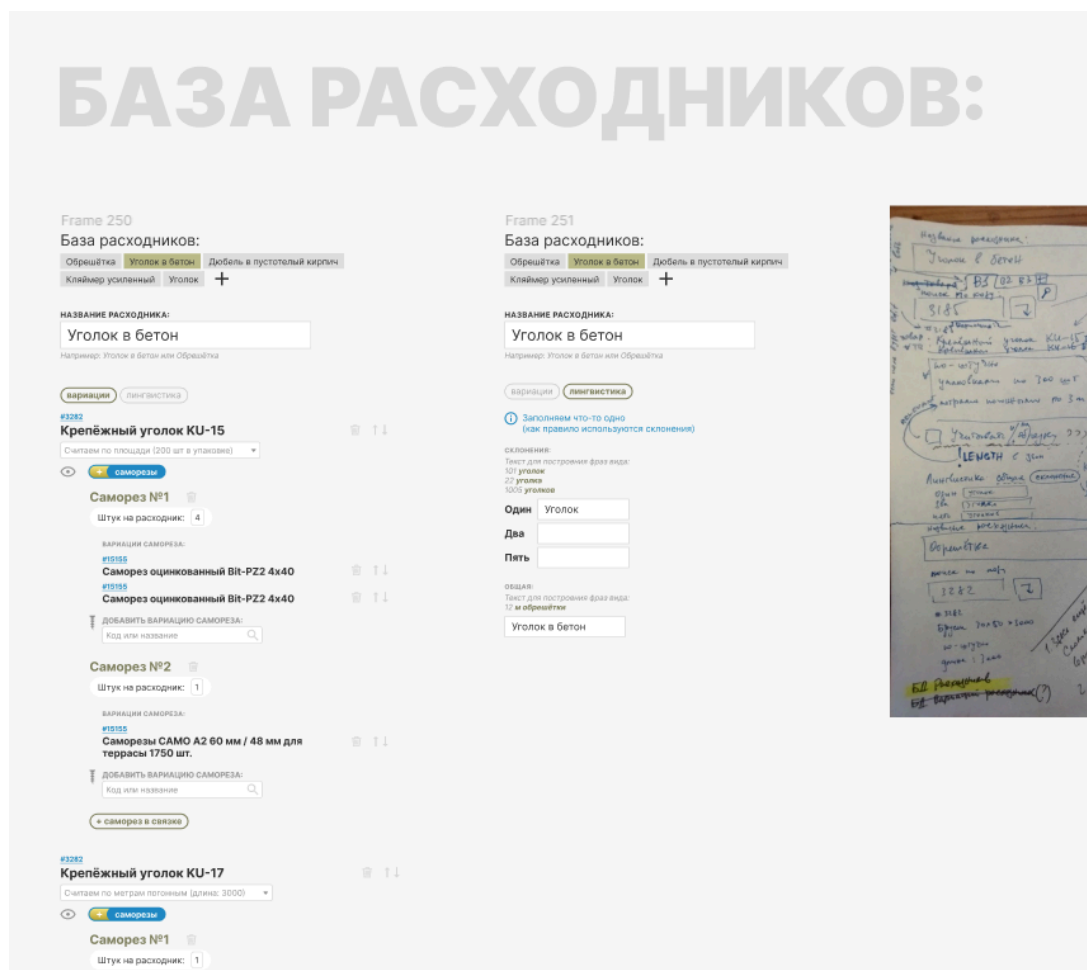


При формировании аякс-запроса передаются в данных ключи:

- условия (rules);
- база наборов (suits);
- формулы (formulas).

Админка расходников

<https://www.figma.com/file/rBWrl0q6V4Med2ZEfuMyvS/%D0%9B%D0%9A%D0%9C-%D0%9A%D0%B0%D0%BB%D1%8C%D0%BA%D1%83%D0%BB%D1%8F%D1%82%D0%BE%D1%80%D0%BE%D0%B2-%D0%90%D0%B4%D0%BC%D0%B8%D0%BD%D0%BA%D0%B0?type=design&node-id=789-349&mode=design&t=hDTsD0g1TCsNuCOZ-0>



При сборке [обрешёток](#) используются справочники расходников (расходники, их вариации, саморезы и их вариации).

Для настройки же самих **расходников** предусмотрена отдельная страница (для саморезов отдельной страницы не предусмотрено, все они не являются самостоятельной сущностью и относятся к определённой вариации расходника).

Заготовка страницы — тут: https://dev.lesobirzha.ru/bitrix/admin/obreshetka_rashodniki_admin.php

Папка с файлами на диске: /local/lib/Calculator/Obreshetka/Admin/

Напоминание: база данных - линейная. [Описано тут](#). Повторяемые товары в вариациях расходника или в вариациях саморезов распределяются по бд столько раз, сколько встречаются. Десять повторов одного и того же товара создают десять разных записей в джсоне (и бд), пусть они по сути и одинаковые.

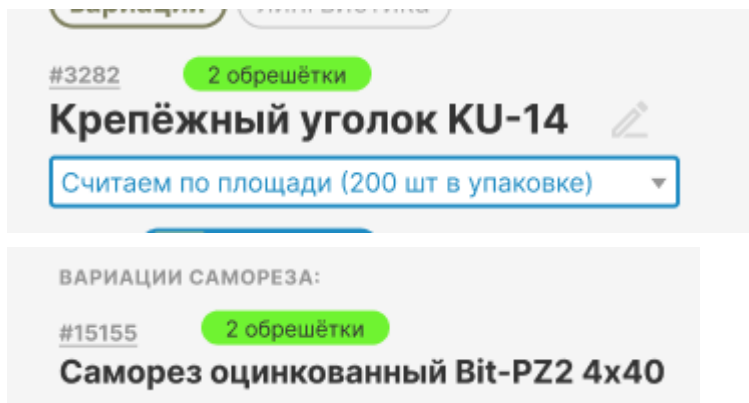
Осторожность

На данной странице мы активно влияем на джсон-ключи:

```
ajax_js: "<script type=\"text/javascript\">1+(!
  vData: {surfacesStruct: {,...},...}
  ▶ calcEntitiesStruct: {calc_obshivka: ["1239",
  ▶ calcStructDb: {calc_obshivka: {NAME: "Калькул
  ▶ consumVariationsDb: {1: {ID: "1", UF_ACTIVE:
  ▶ consumVariationsProducts: {396: {NAME: "Брус
  ▶ consumables: {1: {ID: "1", UF_NAME: "Уголок в
  ▶ entitiesDb: {1239: {CODE: "wall", NAME: "Стен
  ▶ formulaDynamics: {,...}
  ▶ formulas: {1: {ID: "1", UF_FORMULA: "S*5"}, 2
  ▶ formulasCommonMembers: {CPS: {ID: "1", UF_COD
  ▶ formulasUserVariables: {1: {ID: "1", UF_NAME:
  ▶ langs: {betweenKits: "или", ruleIsAlwaysTrue:
  ▶ regions: {6513: {id: 6513, name: "Москва", са
  ▶ rules: {1: [{work_width: {logic: "OR", rules:
  ▶ rulesKeys: {work_width: {ID: "1", UF_NAME: "Ш
  ▶ rulesVariants: {,...}
  ▶ samorezVariationsDb: {1: {ID: "1", UF_SAMOREZ
  ▶ samorezVariationsProducts: {6569: {NAME: "Сам
  ▶ samorezi: {1: {ID: "1", UF_LANG_FOR: "уголок"
  ▶ suits: {,...}
  ▶ surfacesDb: {1336: {NAME: "Обрешетка для карк
  ▶ surfacesStruct: {,...}
  errors: []
```

Необходимо во время разработки постоянно держать в голове, что обрешётка и данная база тесно связаны. То есть не должно допускаться бездумное удаление и редактирование, поскольку расходники могут уже быть задействованы в обрешётках. Лучше несколько раз переспросить или вообще заблокировать доступ к операции, чем предоставить менеджеру инструмент, который может позволить испортить ранее проведённую работу.

В списке расходников публикуем специальные индикаторы, если вариация расходника или вариация самореза уже используется в обрешётке:



Нельзя удалить (скрываем мусорную корзину) и **НЕЛЬЗЯ ДОПУСТИТЬ, ЧТОБЫ ИЗМЕНИЛСЯ КОД (АЙДИ):**

- вариации самореза, если она используется
- вариации расходника, если она используется
- удалить группу Саморез №__, если внутри есть используемая вариация самореза
- расходник, если внутри есть используемая вариация расходника

Добавление

Добавляем расходник через плюстик. Создаётся:

- поле для ввода названия расходника,
- переключатель настроек (вариации, лингвистика),
- поле для добавления вариации расходника (выбора товара из каталога)
- кнопка удаления расходника

База расходников:

Обрешётка Уголок в бетон Дюбель в пустотелый кирпич

Кляймер усиленный Уголок +

НАЗВАНИЕ РАСХОДНИКА:

Уголок в бетон

Например: Уголок в бетон или Обрешётка

вариации лингвистика

ДОБАВИТЬ ВАРИАЦИЮ РАСХОДНИКА

Код или название

Удалить расходник

"Обрешётка"

Основной всей собираемой подсистемы (конструкция, на которую крепится вагонка — под вагонку — подсистема) является обрешётка (деревянный брусок или металлическая полоса). То есть это один из расходников, специально отмеченный. От обрешётки фактически пляшут все формулы, потому что количество всех остальных расходников зависит от метража обрешётки.

В джсоне расходник-обрешётка отмечается ключом UF_BASE

```
▶ 1: {ID: "1", UF_NAME: "Уголок в б
▼ 2: {ID: "2", UF_NAME: "Обрешётка"
  ID: "2"
  SELL_BY_VALUE: "length"
  UF_ACTIVE: "1"
  UF_BASE: "1"
  UF_LANG_CASE_1: ""
  UF_LANG_CASE_2: ""
```

В админке для указания расходника-обрешётки используется флажок в поле ввода названия.

НАЗВАНИЕ РАСХОДНИКА:

Уголок в бетон

Например: Уголок в бетон или Обрешётка

...при установке подсвечивается:

Frame 248

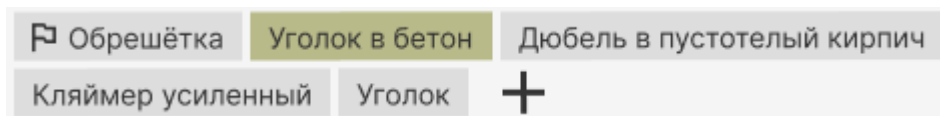
НАЗВАНИЕ РАСХОДНИКА:

Обрешетка



Например: Уголок в бетон или Обрешётка

...а отмеченный расходник отображается на панели с флажочком:



Указать можно только один расходник в качестве обрешётки.

Чтобы установить новый расходник как обрешётку, нужно сначала снять флажок с текущего расходника-обрешётки.

Если ни один расходник не отмечен как обрешётка, отображается большое красное предупреждение (position: fixed):

НЕОБХОДИМО ДОБАВИТЬ РАСХОДНИК-ОБРЕШЕТКУ

Поиск товара в каталоге

Тут и в саморезах: поиск производится динамически, дал пример рабочего кода.

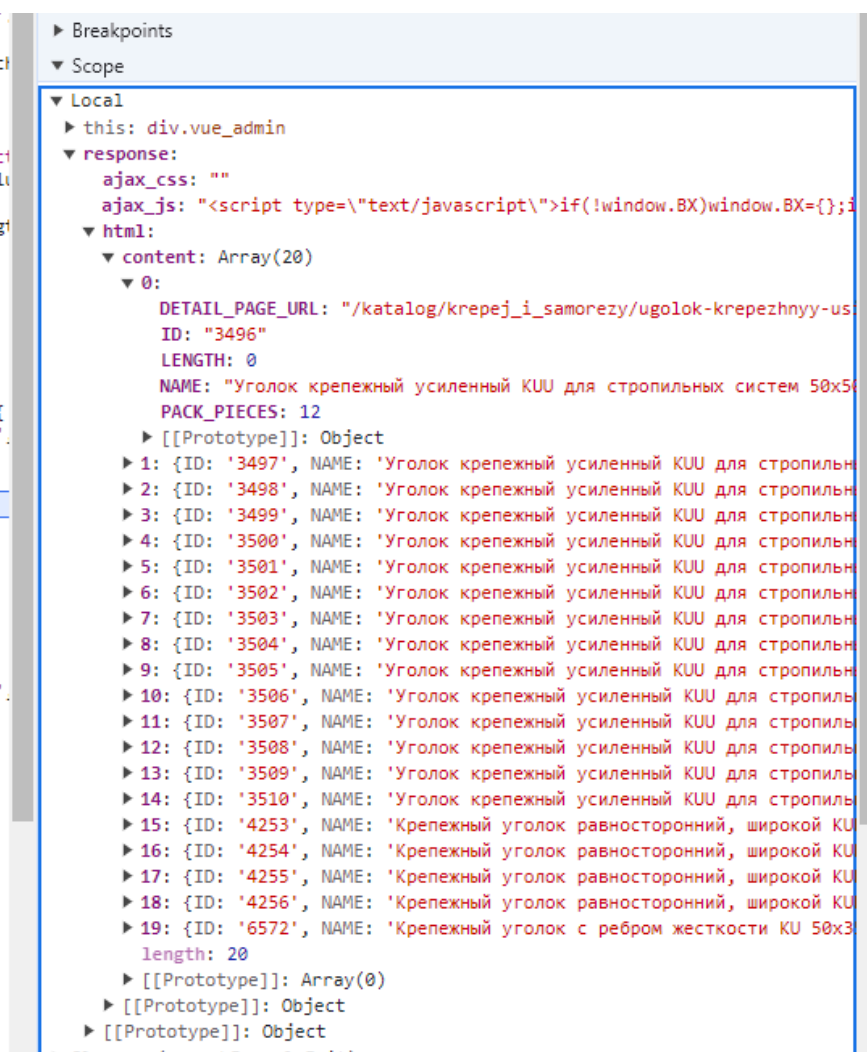
Файлы примера:

- /local/lib/Calculator/Obreshetka/Admin/rashodniki.php — мини-форма

```
// Пример аякс-запроса с поиском
\Bitrix\Main\Page\Asset::getInstance()->addJs
( path: '/local/lib/Calculator/Obreshetka/Admin/ajax_search_example.js' );
?>
<div class="search_example">
<input class="search_text" type="search" value="крен">
<button>Искать</button>
</div>
<?php
```


- /local/lib/Calculator/Obreshetka/Admin/ajax_search_example.js — скрипт аякс-запроса

Аякс-запрос получает в ответ джсон:




#3282

Крепёжный уголок KU-17

Крепёжн 

#3495
Уголок крепежный усиленный KUU для стропильных систем

#3511 
Крепежный уголок KU

#3521
Крепежный уголок KUS под 135 градусов


#3527
Крепежный уголок Z-образный KUZ


1 2 3 4 5

Пример добавленной вариации расходника:

#3282

Крепёжный уголок KU-17

Продаём упаковками (200 шт в упаковке) 

саморезы 


...где:


- **#3282** - код товара + ссылка на товар в каталоге (линк с target="blank"), взято из джсон
- **Крепёжный уголок KU-17** - название товара, взято из джсон
- **Карандашик** открывает интерфейс для перелинковки вариации на другой товар из каталога (будет особенно полезным, когда расходник уже используется в обрешётке но поступил новый товар или предыдущий закончился и вариацию нужно переключить на новый товар)


Frame 248

#3282

Крепёжный уголок KU-17

Продаём упаковками (200 шт в упаковке) 


саморезы 




Frame 279

#3282

Крепёжный уголок KU-17

Код или название 

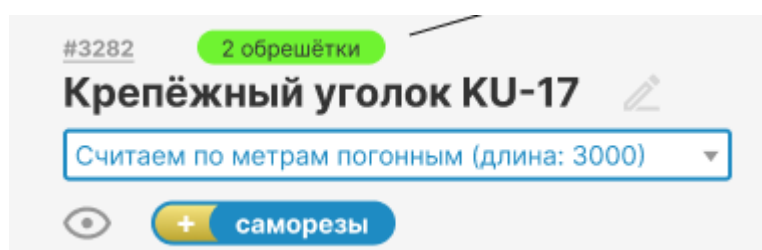
саморезы 

- **Выпадающий список** с настройками продажи/подсчёта - может содержать два варианта, может содержать один. Если вариант один - список disabled.

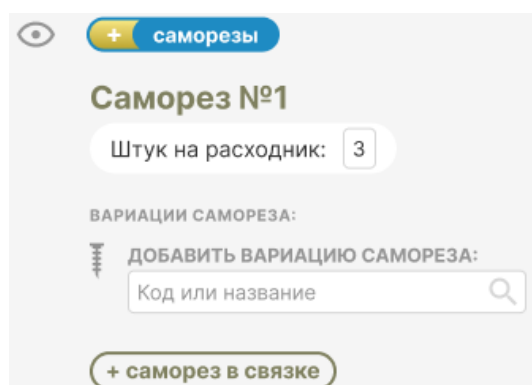
Выбранный в указанном списке параметр попадает в общий джсон в ключ SELL_BY_VALUE

```
▶ consumVariationsProducts: {396: {NAME: "
▼ consumables: {1: {ID: "1", UF_NAME: "Уго
  ▼ 1: {ID: "1", UF_NAME: "Уголок в бетон"
    ID: "1"
    ● SELL_BY_VALUE: "sht"
    UF_ACTIVE: "1"
    UF_LANG_CASE_1: "уголок"
    UF_LANG_CASE_2: "уголка"
    UF_LANG_CASE_5: "уголков"
    UF_LANG_COMMON: ""
    UF_NAME: "Уголок в бетон"
    UF_SORT: "10"
  ▶ 2: {ID: "2", UF_NAME: "Обрешётка", UF_
  ▼ 3: {ID: "3", UF_NAME: "Кляймер", UF_AC
    ID: "3"
    ● SELL_BY_VALUE: "upaks"
    UF_ACTIVE: "1"
```


- Переключатель саморезов. Не все расходники сопровождаются саморезами, поэтому интерфейс с саморезами активируется через переключатель



Саморезы



В подавляющем большинстве те расходники, к которым предусмотрены саморезы, обходятся одним типом саморезов. То есть например на уголок нужно три самореза. В разрезе терминов *саморезы* и *вариации*, под уголок нужна конкретная вариация конкретного самореза в количестве трёх штук.

На скриншоте выше мы видим группу "Саморез №1" с настройкой количества штук и блоком для добавления вариаций на выбор. У названия группы мусорное ведро для удаления не публикуется (потому что группа — одна). Включение переключателя  **саморезы** фактически равносильно созданию группы "Саморез".

Саморез в паре

Другой пример — кляймер. Под него нужен центровой саморез и три боковых. То есть нужны:

- вариация большого самореза, 1 штука
- вариация малого самореза, 3 штуки.

- Записи в джсон вариаций саморезов

Саморез №1

Штук на расходник: 4

ВАРИАЦИИ САМОРЕЗА:

#15155

2 обрешётки

Саморез оцинкованный Bit-PZ2 4x40

#15155

Саморез оцинкованный Bit-PZ2 4x30

ДОБАВИТЬ ВАРИАЦИЮ САМОРЕЗА:

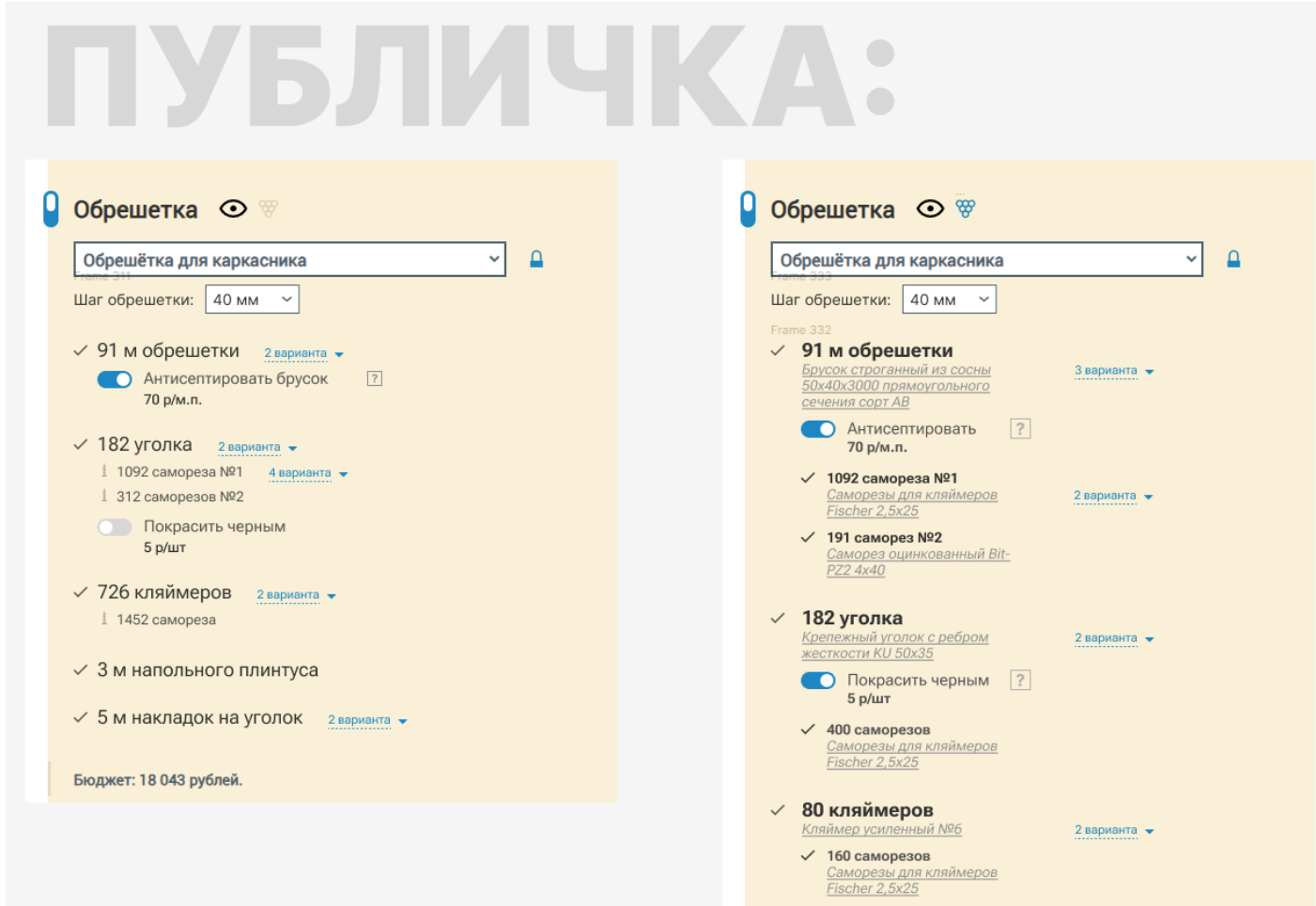
Код или название

```
▼ samorezi: {1: {ID: "1", UF_LANG:  
  ▼ 1: {ID: "1", UF_LANG_FOR: "уг  
    CONSUM_VARIATION_ID: "1"  
    ID: "1"  
    UF_ACTIVE: "1"  
    UF_LANG_FOR: "уголок"  
    UF_QUANTITY: "3"  
  }  
}
```

```
▼ samorezVariationsDb: {1: {ID:  
  ▼ 1: {ID: "1", UF_SAMOREZ_ID:  
    ID: "1"  
    PRODUCT_ID: "25441"  
    UF_ACTIVE: "1"  
    UF_SAMOREZ_I: "1"  
    UF_SORT: "10"
```

Публичка

Дизайн в Фигме.



НОВЫЙ КОМПОНЕНТ

Деревянные поверхности уезжают из хардкода

```
.....<div class="ct_painting_choice">
.....<select
.....    v-model="state.installChoice"
.....    @change="switchInstallChoice"
.....    v-if="entityType == 'wall'"
.....>
.....<option value="">Обрешётка не нужна</option>
.....<option value="karkas">Обрешётка для каркасника</option>
.....<option value="stone">Обрешётка для бетона или полнотелого кирпича</option>
.....<option value="brick">Обрешётка для пустотелого кирпича</option>
.....<option value="concrete">Обрешётка для газобетона</option>
.....<option value="brusbrevno">Обрешётка для брус/бревна</option>
.....</select>
.....<select
.....    v-model="state.installChoice"
.....    @change="switchInstallChoice"
.....    v-else-if="entityType == 'ceil'"
.....>
.....<option value="">Обрешётка не нужна</option>
.....<option value="wood">Обрешётка для дерева</option>
.....<option value="wood_communicate">Обрешётка для дерева с коммуникациями</option>
.....<option value="stone">Обрешётка для плиты (бетон)</option>
.....<option value="stone_communicate">Обрешётка для плиты с коммуникациями</option>
.....</select>
.....<div
.....    v-else
.....>
.....<h1>Ошибка! Неизвестный тип поверхности</h1>
.....</div>
```

...и вся структура динамически управляется через джсоны:

- **surfacesStruct** - структура

```
▼ surfacesStruct: {,..}
  ▼ calc_obshivka: {1239: [1343, 1336, 1337, 1341, 1342], 1240:
    ► 1239: [1343, 1336, 1337, 1341, 1342]
    ► 1240: [1340, 1339, 1344, 1345]
    ► ceil: [1340, 1339, 1344, 1345]
    ► wall: [1343, 1336, 1337, 1341, 1342]
    ► calc_obshivka_spb: {1298: [1347, 1348, 1349, 1350, 1351], wa:
```

- **surfacesDb** - бд поверхностей

```
▼ surfacesDb: {1336: {NAME: "Обрешетка для каркасника", CODE: "kark
  ► 1336: {NAME: "Обрешетка для каркасника", CODE: "karkas", SHORT_I
  ► 1337: {NAME: "Обрешётка для бетона или полнотелого кирпича", COI
  ► 1339: {NAME: "Обрешётка для дерева с коммуникациями", CODE: "woi
  ► 1340: {NAME: "Обрешётка для дерева", CODE: "wood", SHORT_NAME: '
  ► 1341: {NAME: "Обрешётка для пустотелого кирпича", CODE: "brick".
  ► 1342: {NAME: "Обрешётка для газобетона", CODE: "concrete", SHOR
  ► 1343: {NAME: "Обрешётка для брус/бревна", CODE: "brusbrevno", S
```

Все ранее обсуждаемые в админке ключи в публичной части находятся под ключом obreshetka:

```
▼ obreshetka: {surfacesStruct: {,...},...}
  ▶ consumVariationsDb: {1: {ID: "1", UF_ACTIVE: "1", UF_CONSUMABLE
  ▶ consumVariationsProducts: {396: {NAME: "Брусек строганный из со
  ▶ consumables: {,...}
    formulaAdminDefault: "D_OBRPM*4"
  ▶ formulaDynamics: {,...}
  ▶ formulas: {1: {ID: "1", UF_FORMULA: "S*5"}, 2: {ID: "2", UF_FOR
  ▶ formulasCommonMembers: {CPS: {ID: "1", UF_CODE: "CPS", UF_SORT:
  ▶ formulasUserVariables: {,...}
  ▶ processings: [{UF_NAME: "Антисептирование бруска", UF_CODE: "an
  ▶ rules: {1: [{work_width: {logic: "OR", rules: [{COND: "=", VALU
  ▶ rulesKeys: {work_width: {ID: "1", UF_NAME: "Ширина доски", UF_C
  ▶ samorezVariationsDb: {1: {ID: "1", UF_SAMOREZ_ID: "1", UF_SORT:
  ▶ samorezVariationsProducts: {6569: {NAME: "Саморез оцинкованный
  ▶ samorezi: {1: {ID: "1", UF_LANG_FOR: "уголок", UF_ACTIVE: "1",
  ▶ suits: {,...}
  ▶ surfacesDb: {1336: {NAME: "Обрешетка для каркасника", CODE: "ka
  ▶ surfacesStruct: {,...}
```

Концепция

Компонент в калькуляторе представляет собой модификацию текущего компонента, по крайней мере внешне мы ориентируемся на то, что уже сделано. То есть за основу берём то, что сейчас, делаем копию, например WallsMontazhExt2023 (чтобы одновременно была возможность использовать и старую версию и новую) и расширяем инструментами, которые уже реализованы в расходниках.

Новый компонент полностью перезажает на новые джсоны:

```
▼ wallsMontazhExt2023: {,...}
  ▶ calcEntitiesStruct: {calc_obshivka: ["1239", "1240"
  ▶ calcStructDb: {calc_obshivka: {NAME: "Калькулятор о
  ▶ consumVariationsDb: {1: {ID: "1", UF_ACTIVE: "1", U
  ▶ consumVariationsProducts: {396: {NAME: "Брусек стро
  ▶ consumables: {1: {ID: "1", UF_NAME: "Уголок в бетон
  ▶ entitiesDb: {1239: {CODE: "wall", NAME: "Стена"}, 1
  ▶ formulaDynamics: {,...}
  ▶ formulas: {1: {ID: "1", UF_FORMULA: "S*5"}, 2: {ID:
  ▶ formulasCommonMembers: {CPS: {ID: "1", UF_CODE: "CP
  ▶ formulasUserVariables: {1: {ID: "1", UF_NAME: "Шар
  ▶ langs: {betweenKits: "или", ruleIsAlwaysTrue: "Отбр
  ▶ regions: {6513: {id: 6513, name: "Москва", calculat
  ▶ rules: {1: [{work_width: {logic: "OR", rules: [{CON
  ▶ rulesKeys: {work_width: {ID: "1", UF_NAME: "Ширина
  ▶ rulesVariants: {,...}
  ▶ samorezVariationsDb: {1: {ID: "1", UF_SAMOREZ_ID: "
  ▶ samorezVariationsProducts: {6569: {NAME: "Саморез о
  ▶ samorezi: {1: {ID: "1", UF_LANG_FOR: "уголок", UF_A
  ▶ suits: {,...}
  ▶ surfacesDb: {1336: {NAME: "Обрешетка для каркасника
  ▶ surfacesStruct: {,...}
```

Переменные

Для проверки выполнения условий необходимо получать значения используемых параметров (ширина доски, толщина доски, используемый профиль). Источники данных указаны в джсоне, описывающем ключи условий, в параметре UF_RELATIVE_CHAIN:

```

▼ rulesKeys: {,...}
  ▼ matherial: {ID: "3", UF_NAME: "Материал", UF_CODE: "matheri
    ID: "3"
    TYPE_CODE: "list"
    UF_CODE: "matherial"
    UF_CUSTOM_ORIGIN: "catalog:matherial"
    UF_ENUM_ORIGIN: ""
    UF_NAME: "Материал"
    UF_RELATIVE_CHAIN: "product:IBLOCK_SECTION_CODE"
  ▼ wood_thickness: {ID: "2", UF_NAME: "Толщина доски", UF_CODE
    ID: "2"
    TYPE_CODE: "number"
    UF_CODE: "wood_thickness"
    UF_CUSTOM_ORIGIN: ""
    UF_ENUM_ORIGIN: ""
    UF_NAME: "Толщина доски"
    UF_RELATIVE_CHAIN: "D:woodThickness"
  ► work_width: {ID: "1", UF_NAME: "Ширина доски", UF_CODE: "wc

```

...где через ":" указан источник и название переменной.

Источник "D" — любая из [переменных](#). Потому на момент анализа условий все переменные уже должны быть проинициализированы.


Источник "product" — переменная берётся из выбранного товара

```



▼ products: Object
  ▼ 10898: Object
    CODE: "vagonka-iz-angarskoy-sosny-sl
    IBLOCK_SECTION_CODE: "vagonka"
    IBLOCK_SECTION_ID: "96"
    ID: "10898"
    NAME: "Вагонка из ангарской сосны ш
  ► PACK: Object
    PIECES_IN_PACK: "10"
  ► SINGLE: Object
    URL: "/katalog/vagonka/vagonka-iz-a
  ► USE_IN_OUT: Array[0]
    WOOD_PROFIL_CODE: "shstil"
    WOOD_PROFIL_ID: "47"
    WOOD_PROFIL_VALUE: "Штиль"
    WOOD_THICKNESS_VALUE: "14"
    WOOD_WIDTH_VALUE: "115"

```

Расходник-обрешетка



Обрешетка

Обрешётка для каркасника

▼

🔒

Шаг обрешетки:

40 мм

▼

✓ 91 м обрешетки

2 варианта ▼

☒

Антисептировать брусok

?

70 р/м.п.

✓ 182 уголка

2 варианта ▼

Расходник, отмеченный как обрешётка (ключ UF_BASE)

```
▼ consumables: {,...}
  ► 1: {ID: "1", UF_NAME: "Уголок в бетон",
  ▼ 2: {ID: "2", UF_NAME: "Обрешётка", UF_
    ID: "2"
    SELL_BY_VALUE: "length"
    UF_ACTIVE: "1"
    UF_BASE: "1"
    UF_LANG_CASE_1: ""
```

...должен быть всегда первым в списке.

"Антисептирование"

У расходников могут быть активированы дополнительные услуги (в текущей версии - для бруска обрешётки, но в будущем ничего исключать нельзя), потому что джсон составлен так, словно дополнительные услуги могут быть у любого из расходников.

Расчёт стоимости производим по текущему значению расходника, к которому привязана каждая услуга (если услуга для обрешётки, значит цену считаем по текущему значению обрешётки; если для плинтуса — берём текущее значение плинтуса. Если для уголков — количество уголков...

✓ 91 м обрешетки 2 варианта ▼
☒ Антисептировать брусок 70 р/м.п. [?]
✓ 182 уголка 2 варианта ▼
1092 самореза №1 4 варианта ▼
312 саморезов №2
☐ Покрасить черным 5 р/шт
✓ 726 кляймеров 2 варианта ▼
1452 самореза

Джсон — в ключе processings:

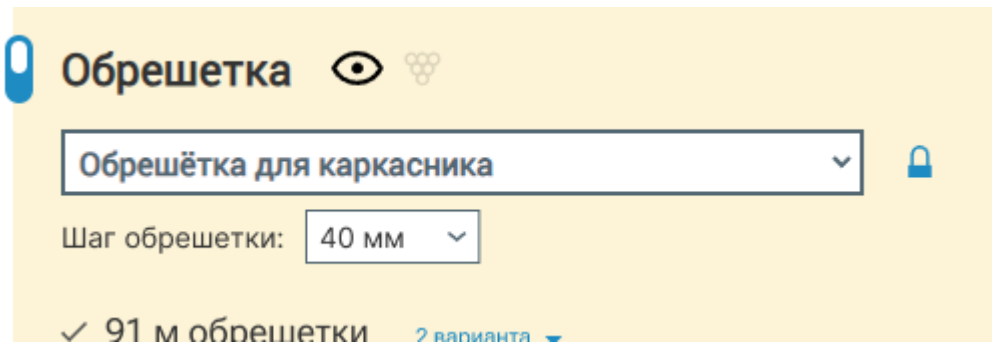
```
▼ processings: [{UF_NAME: "Антисептирование бруска", UF_CODE:
  ▼ 0: {UF_NAME: "Антисептирование бруска", UF_CODE: "antiseptirovanie"
    UF_CODE: "antiseptirovanie"
    ► UF_CONSUMABLE_ID: ["1", "2"]
    UF_DESC: "Защита дерева с помощью безопасного для чело
    UF_MEASURE: "м.п."
    UF_NAME: "Антисептирование бруска"
    UF_PRICE: "75"
```

...где UF_CONSUMABLE_ID - массив расходников, для которых данная услуга действует.

!!! Привязку (активированные пользователем услуги) производим по коду (ключ UF_CODE).

Шаг обрешетки

Вводится настройка "Шаг обрешётки".



Выпадающий список — набор значений, построенных в диапазоне "от и до" с установленным шагом. Настройки для построения списка значений берутся из джсона `formulasUserVariables`, из ключа `OBR_STEP`:

```

▼ formulasUserVariables: {,...}
  ▼ OBR_STEP: {ID: "1", UF_NAME: "Шаг обрешётки",
    ID: "1"
    UF_CODE: "OBR_STEP"
    UF_DEFAULT: "30"
    UF_DIA_FROM: "20"
    UF_DIA_STEP: "10"
    UF_DIA_TO: "150"
    UF_NAME: "Шаг обрешётки"
    ► UF_VARIANTS: [20, 30, 40]
  }
}

```

Возможно два варианта использования джсона:

1. Задан UF_VARIANTS — список строится по нему
2. UF_VARIANTS не задан, тогда список строится от UF_DIA_FROM до UF_DIA_TO с шагом UF_DIA_STEP

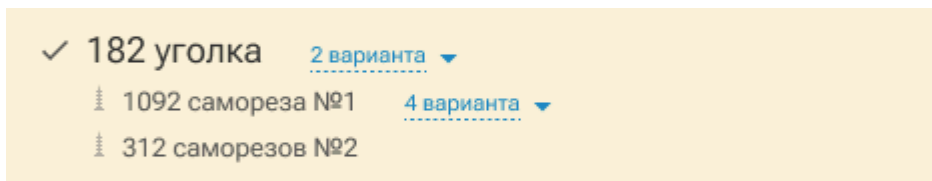
В обоих случаях значение по-умолчанию берётся из ключа UF_DEFAULTS.

Вариации расходников

По примеру расходников из калькулятора покраски / герметизация швов, в обрешётку завозим вариации расходников (как самих расходников так и саморезов к ним)

Саморезы в паре

Помним про обробку ситуації "Саморезы в паре"



Режим виногради́ка

Новый дизайн.

Обрешетка

Обрешётка для каркасника

Шаг обрешетки:

40 мм

✓

91 м обрешетки

Брусок строганный из сосны

50x40x3000 прямоугольного сечения сорт АВ

3 варианта

Antisepticheskoye sredstvo

Antisepticheskoye sredstvo

Antisepticheskoye sredstvo

Antisepticheskoye sredstvo

✓

1092 самореза №1

Саморезы для кляймеров Fischer 2,5x25

2 варианта

✓

191 саморез №2

Саморез оцинкованный Bit-PZ2 4x40

✓

182 уголка

Крепежный уголок с ребром жесткости KU 50x35

2 варианта

Покрасить черным

Покрасить черным

Покрасить черным

Покрасить черным

✓

400 саморезов

Саморезы для кляймеров Fischer 2,5x25

Неочевидное

Не потерять по дороге то, что уже сделано

Галочки с выбором расходников на покупку

Обрешетка

Обрешётка для брус/бревна

202 м обрешётки

1452 кляймеров

2904 самореза под кляймеры

404 опор

808 саморезов под опоры

Нет ширины и высоты

Плинтус напольный

Не указаны необходимые для расчётов ширина и высота

Накладка на уголок

Не указаны необходимые для расчётов ширина и высота

Актуально для расходников, которые считаются метрами погонными.

Здесь у нас конечно логические ножницы, потому что мы указываем тип расчёта у *вариаций расходников*, а в публичной части фактически публикуется логическая группа вариаций — *расходник*. Тяжело предсказать, какова окажется реальность, но исходим из того, что: вариаций будет много, однако в плане типа расчета они будут одинаковы.

А значит проверку расходника на чувствительность к режиму ш*в проводим так: если хотя бы одна вариация расходника считается метрами погонными, весь расходник отмечаем чувствительным к режиму ш*в.