

Средства среды разработки моделей

Валидация ошибок

```
set duration() {
    return длительность_обслуживания.next(
        _Парикмахер.длительность_min,
        _Парикмахер.длительность_max
    );
}

set duration() {
    return x + 5;
}
```

Error - default method cannot be set more than once
Press 'F2' for focus

Навигация по коду модели

```
set duration() {
    return длительность_обслуживания.next(_Парикмахер.длительность_min,
}

set begin() {
    _Парикмахер.состояние_парикмахера = Состояние_парикмахера.Занят;
}

set end() {
    _Парикмахер.состояние_парикмахера = Состояние_парикмахера.Свободен;
    _Парикмахер.количество_обслуженных = _Парикмахер.количество_обслуженных + 1;
    _Клиент.erase();
}
```

Uniform clients.длительность_обслуживания
Original declaration:
Sequence длительность_обслуживания
Press 'F2' for focus

Автоматическое переименование

```
set begin() {
    _Парикмахерская.количество_в_очереди = _Парикмахерская.количество_в_очереди - 1;
    _Клиент.состояние = Состояние_клиента.Начал_стрижку;
    _Парикмахер.состояние_парикмахера = Состояние_парикмахера.Занят;
}

set end() {
    _Парикмахер.состояние_парикмахера = Состояние_парикмахера.Свободен;
    _Парикмахер.количество_обслуженных = _Парикмахер.количество_обслуженных + 1;
    _Клиент.erase();
}
```

Enter new name, press Enter to refactor

Подсветка синтаксиса

```
operation образец_обслуживания_клиента() {
    relevant _Парикмахерская = парикмахерская.onlyif[
        количество_в_очереди > 0
    ];
    relevant _Клиент = Клиенты.all.filter[состояние.equals(
        Состояние_клиента.Пришел
    )].any;
    relevant _Парикмахер = Парикмахеры.all.filter [
        состояние_парикмахера.equals(Состояние_парикмахера.Свободен)
        && тип_клиента.equals(_Клиент.тип)
    ].minBySafe[количество_обслуженных];

    set duration() {
        return длительность_обслуживания.next(
            _Парикмахер.длительность_min,
            _Парикмахер.длительность_max
        );
    };

    //
    // set duration() {

```

Автодополнение кода

```
operation образец_обслуживания_клиента() {
    relevant _Парикмахерская = парикмахерская.onlyif[количество_в_очереди > 0];
    relevant _Клиент = Клиенты.all.filter[состояние.equals(Состояние_клиента.Пришел)].any;
    relevant _Парикмахер = Парикмахеры.all.filter[состояние_парикмахера.equals(Состояние_парикмахера.Свободен)
        && тип_клиента.equals(_Клиент.тип)].minBySafe[к

    set int Парикмахеры.получитьКоличество_обслуженных()
    }
    set
    }
    set
}
```

Original declaration:
FieldDeclaration
количество_обслуженных: int - Парикмахеры.получитьКоличество_обслуженных
количество_обслуженных = value: void - Парикмахеры.получитьКоличество_обслуженных
Ctrl+Space to show shortest proposals

Автоматическое форматирование

```
sequence интервал_прихода = new Exponential(123456789, 1/30.0);
sequence длительность_обслуживания = new Uniform(123456789);
sequence случайный_тип_клиента = new DiscreteHistogram
<Тип_клиента>(123456789,
    #[Тип_клиента.Tun1 -> 1.0,
    Тип_клиента.Tun2 -> 5.0
]);

sequence интервал_прихода = new Exponential(123456789, 1/30.0);
sequence длительность_обслуживания = new Uniform(123456789);
sequence случайный_тип_клиента = new DiscreteHistogram<Тип_клиента>(123456789,
    #[
        Тип_клиента.Tun1 -> 1.0,
        Тип_клиента.Tun2 -> 5.0
    ]
);
```