

Автоматизация тестирования Web UI на Java

# Selenium WebDriver

[Java 11]

---



# На этом уроке

1. Узнаем, что такое Selenium WebDriver.
2. Познакомимся с Selenium IDE.
3. Поговорим о создании Test Project и Test Case.
4. Научимся создавать проекты.

## Оглавление

[Исторический экскурс](#)

[Selenium WebDriver](#)

[Selenium IDE](#)

[Создание Test Project и Test Case](#)

[Практическое задание](#)

[Тест-кейс 1. Создание проекта](#)

[Тест-кейс 2. Создание контактного лица в организации с минимально заполненной информацией](#)

# Исторический экскурс

Selenium — это проект, в рамках которого разрабатывается серия программных продуктов с открытым исходным кодом (Open source):

1. Selenium WebDriver.
2. Selenium IDE.
3. [Selenium RC](#).
4. [Selenium Server](#).
5. [Selenium Grid](#).

В нашем курсе мы поговорим о первых двух. Эти программные продукты покрывают практически все потребности автотестировщика.

## Selenium WebDriver

Selenium WebDriver — это программная библиотека для управления браузерами. Употребляется также более короткое название — WebDriver.

Иногда эту библиотеку называют «драйвер браузера». Однако это целое семейство драйверов для различных браузеров, а также набор клиентских библиотек на разных языках, позволяющих работать с этими драйверами. Это основной продукт, разрабатываемый в рамках проекта Selenium.

Selenium WebDriver называется также Selenium 2.0.

Selenium 1.0 (RC) + WebDriver = Selenium 2.0.

По сравнению с Selenium RC WebDriver использует совершенно иной способ взаимодействия с браузерами. Он напрямую вызывает команды браузера, используя родной для каждого конкретного браузера API. Как совершаются эти вызовы и какие функции они выполняют зависит от конкретного браузера. В то же время Selenium RC внедрял JavaScript-код в браузер при запуске и использовал его для управления веб-приложением. Таким образом, WebDriver использует способ взаимодействия с браузером более близкий к действиям реального пользователя.

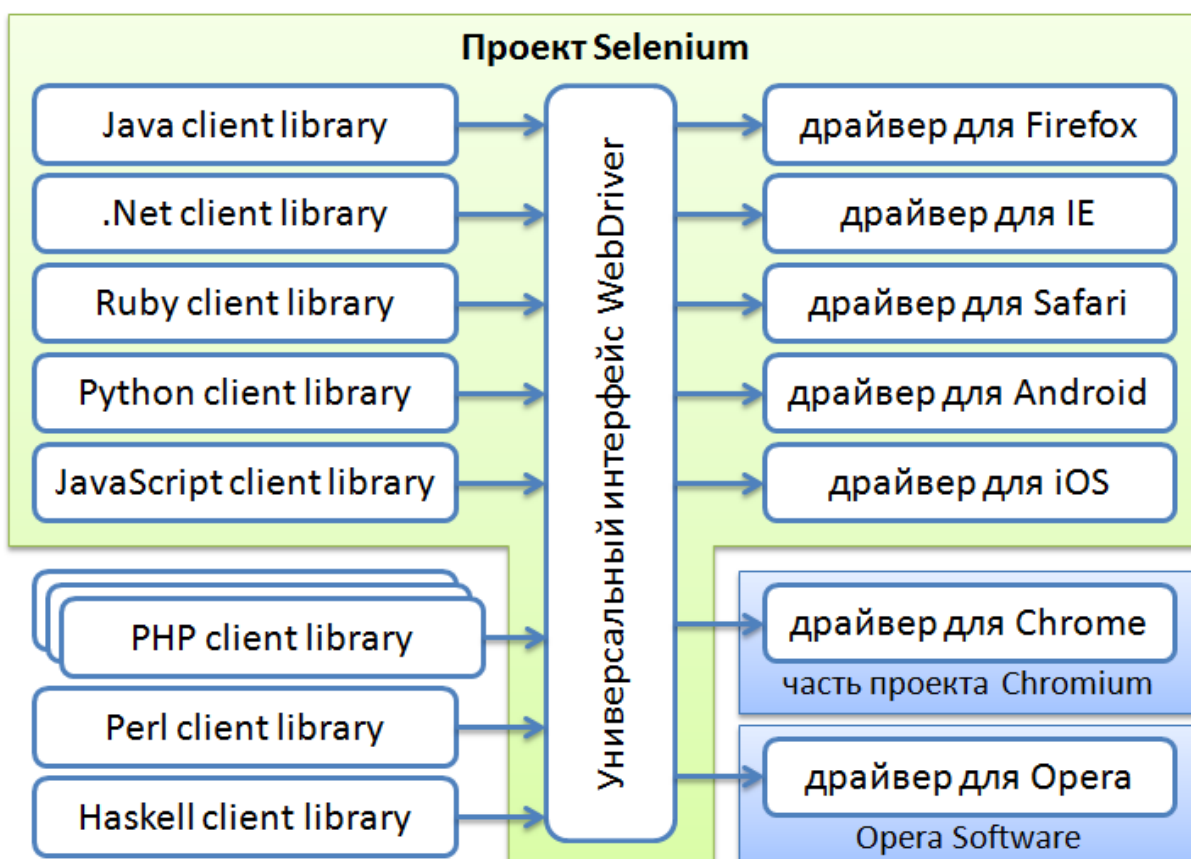
По сравнению с более старым интерфейсом он обладает рядом преимуществ:

1. Интерфейс WebDriver спроектирован более простым и выразительным.
2. WebDriver обладает более компактным и объектно-ориентированным API.
3. WebDriver управляет браузером более эффективно, а также справляется с некоторыми ограничениями, характерными для Selenium RC, как:
  - загрузка и отправление файлов;
  - попапы;
  - диалоги.

Для работы с WebDriver требуются 3 основных программных компонента:

1. Браузер, работу которого пользователь хочет автоматизировать. Это реальный браузер конкретной версии, установленный на определённой ОС и имеющий свои настройки — по умолчанию или кастомные. Однако WebDriver работает и с «ненастоящими» браузерами, но подробно о них позже.
2. Для управления браузером требуется driver браузера. Driver — веб-сервер. Он запускает браузер и отправляет ему команды, а также закрывает его. У каждого браузера свой driver. Связано это с тем, что у каждого браузера собственные отличные команды управления и реализованы они по-своему. Найти список доступных драйверов и ссылки для скачивания можно [на официальном сайте Selenium](#) проекта.
3. Скрипт — это тест. Он содержит набор команд на конкретном языке программирования для драйвера браузера. Такие скрипты используют Selenium WebDriver bindings (готовые библиотеки). Эти библиотеки доступны пользователям на различных языках. В рамках проекта Selenium разрабатываются библиотеки для языков Java, .NET (C#), Python, Ruby, JavaScript. Все остальные реализации не имеют отношения к проекту Selenium, хотя, возможно, в будущем, какие-то из них волеются в этот проект.

WebDriver представляет собой семейство драйверов для различных браузеров с набором клиентских библиотек для этих драйверов на разных языках программирования:



В рамках проекта Selenium разрабатываются драйверы для браузеров Firefox, Internet Explorer и Safari, а также драйверы для мобильных браузеров Android и iOS. Драйвер для браузера Google Chrome создаётся в рамках проекта Chromium, а драйвер для браузера Opera (включая мобильные версии) — компанией Opera Software. Поэтому формально они — не часть проекта Selenium, распространяются и поддерживаются независимо. Но логически, эти драйверы относятся к семейству продуктов Selenium.

## Selenium IDE

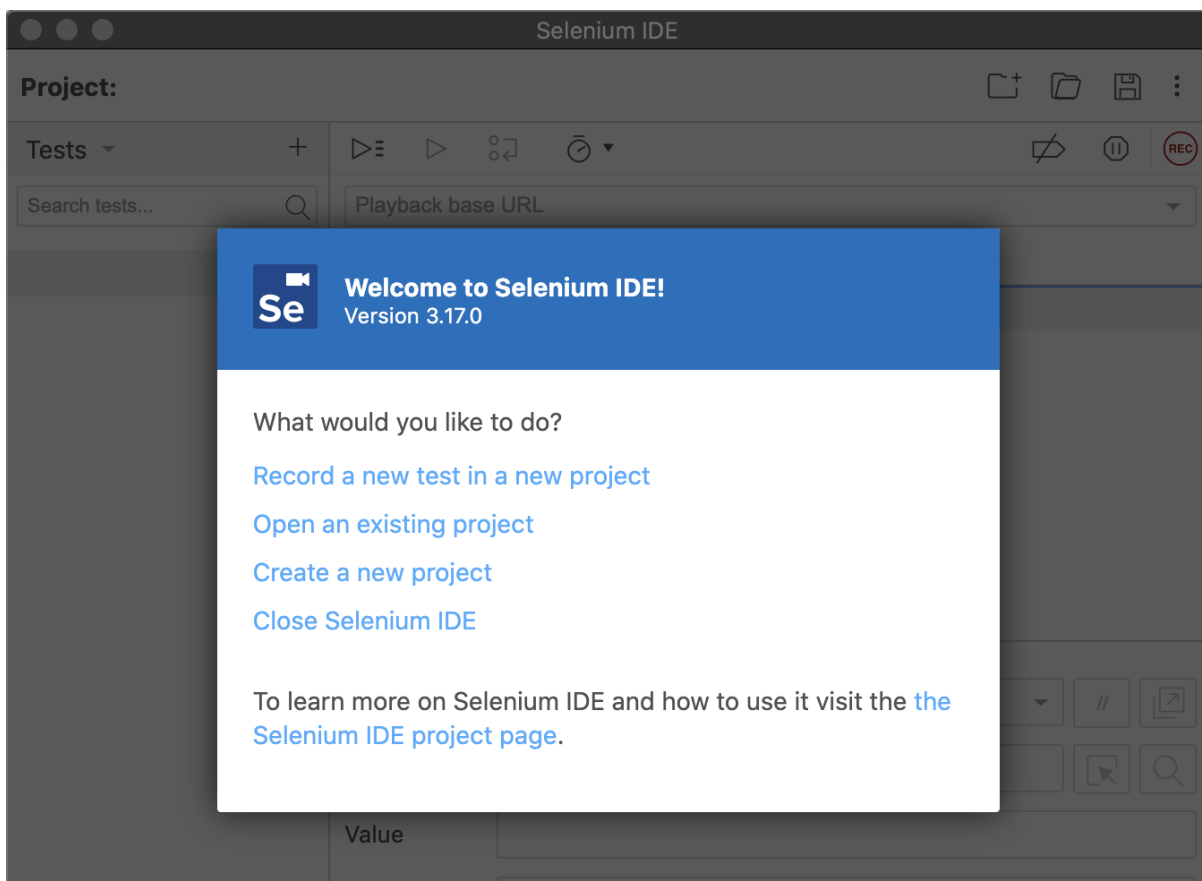
Изначально Selenium IDE — это плагин для браузера Mozilla FireFox, но сейчас он предоставляется усилиями сообщества ещё и для Google Chrome. Скачивается он по [этой ссылке](#) или в менеджере расширений браузера.

В [документации](#) написано, что после перехода на страницу скачивания браузер сам предложит установить плагин и попросит разрешения. Если же такого не произошло, то нужно проскроллить страницу до подзаголовка Selenium IDE и нажать на ссылку, следующую за словами Download latest released version. После этого браузер выводит окно для подтверждения установки плагина. Соглашаемся и устанавливаем его, затем перезапускаем браузер. У вас появится новая иконка в правом углу панели браузера.



## Создание Test Project и Test Case

Откроем окно Selenium IDE в браузере и нажмём на его иконку в панели. Появится следующее окно:



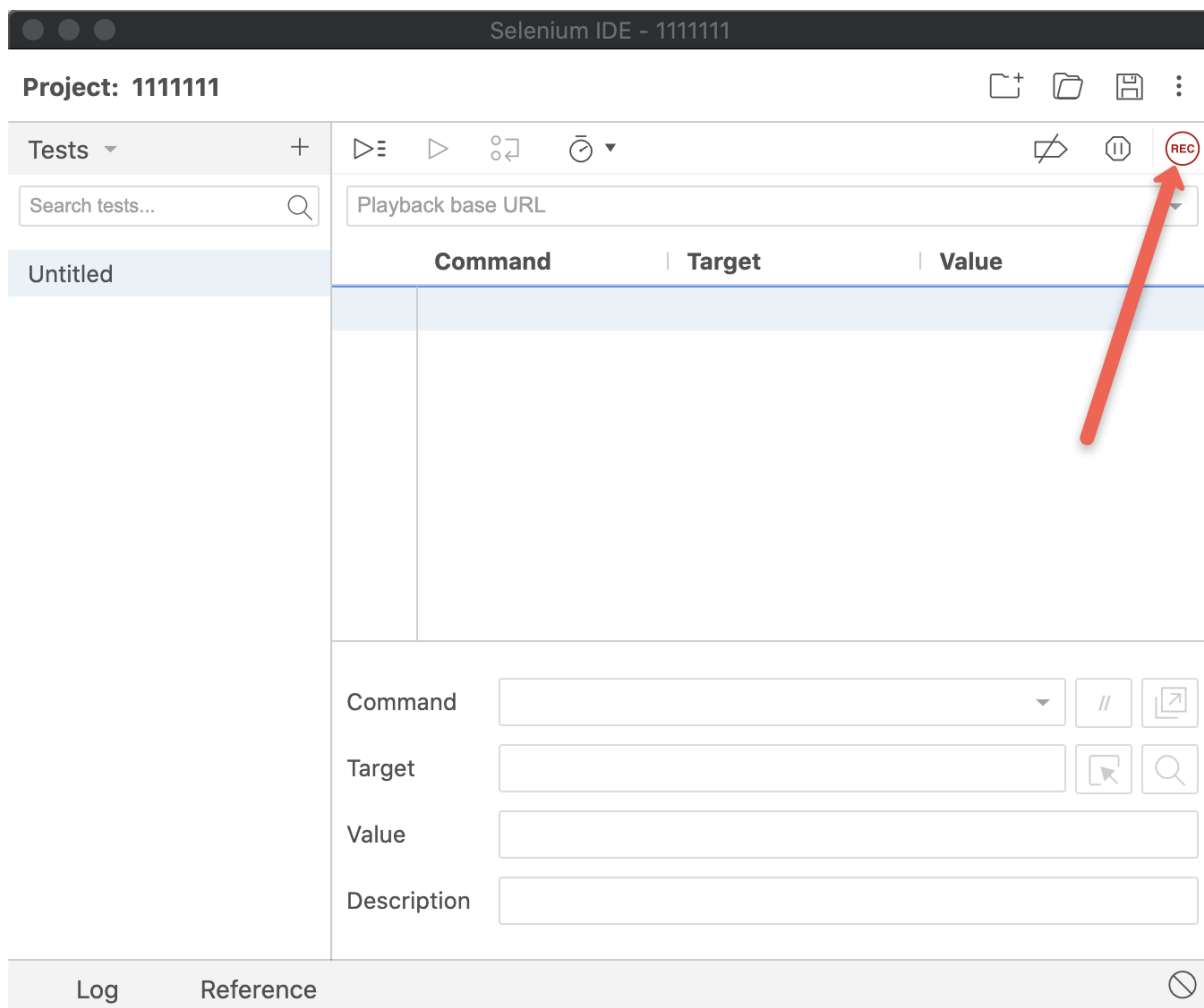
В Selenium IDE есть два основных структурных элемента:

1. **Test (ранее — Test Case).**
2. **Test Project (ранее — Test Suite).**

**Test** — это набор команд. Они указывают Selenium IDE, как проводить тестирование сайта. Чуть ниже описывается создание таких команд для решения поставленных задач по проверке телефона и услуги на разных страницах.

**Test Project** — это набор из нескольких Testов. Призван для логической группировки тест-кейсов в одном месте.

Для начала выберем опцию Create a new project. Дадим название проекту и нажмём на кнопку записи в появившемся окне:

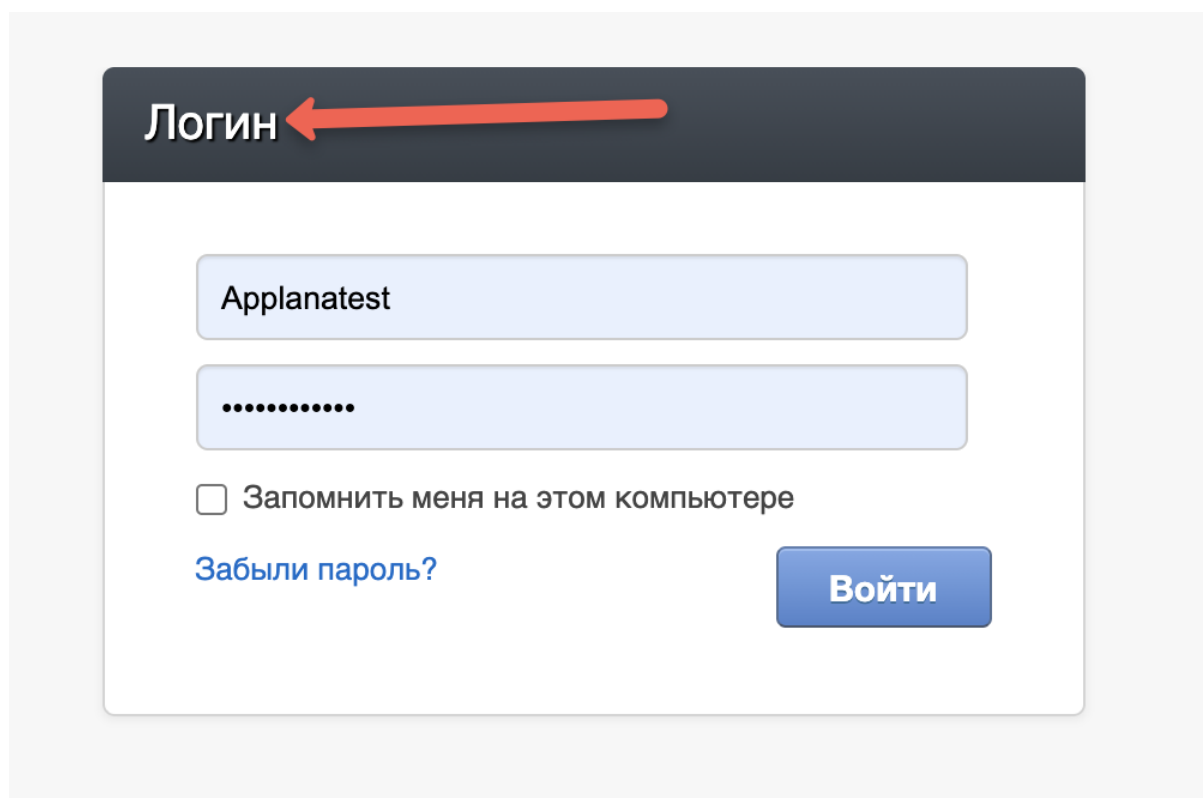


Появится диалоговое окно ввода тестируемого сайта. Введём базовый URL без подразделов, например, этот.

Selenium IDE после этого откроет новое окно браузера с указанным сайтом. С этого момента Selenium IDE записывает все наши действия с сайтом: клики мышкой, скроллы, заполнение полей. Об этом свидетельствует активированная красная кнопка записи в правом углу окна. Сделав любое действие, например, авторизовавшись в системе, мы увидим, как это запишет Selenium IDE в шагах теста:

<div> <span>▶</span> <span>▶</span> <span>🔄</span> <span>⌚</span> <span>✂</span> <span>⏸</span> <span>REC</span> </div> <div>https://crm.geekbrains.space/</div>		
	Command	Target   Value
1	open	https://crm.geekbrains.space/user/login
2	set window size	1603x1017
3	click	id=prependInput

После остановки записи записанный тест проигрывается по кнопке Play. Но для начала добавим проверку на текст внутри элемента заголовка окна:



Для этого поставим курсор после шага 3 и начнём вводить assert text в поле Command. Автозаполнение при этом поможет нам выбрать нужную опцию. Затем укажем в поле Target адрес поля. Воспользуемся кнопкой Select target on page:



Command

Target



Далее нужно ввести текст в поле Value. С этим полем начнёт сравниваться значение на странице, в нашем случае — «Логин».

Запустим тест по команде Run current test. Автоматически возникнет новое окно браузера, и мы увидим ход выполнения теста.

В Selenium IDE есть возможность наблюдать за ходом выполнения. Пройденные шаги окрасятся в зелёный, шаг, повлёкший ошибку — в красный:

Selenium IDE - 1111111\*

Project: 1111111\*

Executing ▾

Test Edit Project\*

https://crm.geekbrains.space/

	Command	Target	Value
1	open	https://crm.geekbrains.space/user/login	
2	set window size	1603x1017	
3	click	id=prependInput	
4	assert text	css=.title	Логин

Command

Target

Value

Description

Runs: 1 Failures: 0

Log Reference


1. open on https://crm.geekbrains.space/user/login OK 16:48:05


2. setWindowSize on 1603x1017 OK 16:48:05

3. click on id=prependInput OK 16:48:05

4. assertText on css=.title with value Логин OK 16:48:08

'Test Edit Project' completed successfully 16:48:08





Тест с ошибкой:

Project: 1111111\*

Executing ▾

Test Edit Project\*

https://crm.geekbrains.space/

	Command	Target	Value
1	open	https://crm.geekbrains.space/user/login	
2	set window size	1603x1017	
3	click	id=prependedInput	
4	assert text	css=.title	fgdgdgdfdfgdfg

Runs: 1 Failures: 1

Log Reference

Running 'Test Edit Project' 16:59:28

1. open on https://crm.geekbrains.space/user/login OK 16:59:29

2. setWindowSize on 1603x1017 OK 16:59:29

3. click on id=prependedInput OK 16:59:29

4. assertText on css=.title with value fgdgdgdfdfgdfg Failed: 16:59:32  
Actual value "Логин" did not match "fgdgdgdfdfgdfg"

'Test Edit Project' ended with 1 error(s) 16:59:32

Отлично, теперь мы уверены, что на сайте отображается правильный заголовок формы. Сохраним наши наработки — нажмём Ctrl+S и сохраним файл тест-кейса в отдельную папку. Стараемся давать осмысленные названия файлам — так легче понять, что именно они проверяют.

## Практическое задание

1. Автоматизируйте сценарии создания проекта и контактного лица в [CRM](#). Добавьте тесты в один проект и сохраните его в формате .side.
2. Выберите любой публичный сайт, не попадающий под NDA, который используете в течение всего курса. Автоматизируйте два любых сценария, например, авторизация на [сайте](#) и создание новой записи. Абсолютный минимум — один тест на каждый позитивный сценарий (без негативных проверок). Сохраните их в другой проект и также сдайте в .side-формате.

# Тест-кейс 1. Создание проекта

Условия:

1. Есть организация, в которой есть хотя бы одно контактное лицо. Для создания контактного лица в имеющийся организации см. ТК-2.
2. В организации есть следующие объекты: Подразделение, Куратор проекта, Руководитель проекта, Администратор проекта, Менеджер. По умолчанию последние присутствуют в системе.

№	Шаг	Ожидаемый результат
1	Авторизоваться на сайте CRM, используя следующие данные: <a href="#">URL</a> ; Логин/пароль: Applanatest1/Student2020!	Пользователь успешно авторизовался, видит страницу «Панель инструментов».
2	Перейти в «Проекты» → «Мои проекты»	Пользователь находится на странице «Мои проекты», присутствует кнопка «Создать проект».
3	Нажать на кнопку «Создать проект»	Открыта страница создания проекта.
4	Заполнить обязательные поля: <ul style="list-style-type: none"><li>• наименование;</li><li>• организация;</li><li>• основное контактное лицо;</li><li>• подразделение;</li><li>• куратор проекта;</li><li>• руководитель проекта;</li><li>• администратор проекта;</li><li>• менеджер.</li></ul>	Поля заполнены.
5	Нажать на кнопку «Сохранить и закрыть»	Страница создания проекта закрыта. Пользователь видит страницу «Все проекты» и всплывающее уведомление о том, что проект успешно создан.

# Тест-кейс 2. Создание контактного лица в организации с минимально заполненной информацией

Условия:

3. В системе есть хотя бы одна организация. Список организаций находится в «Контрагенты» → «Организации».

№	Шаг	Ожидаемый результат
1	Авторизоваться на сайте CRM, используя следующие данные: <a href="#">URL</a> ; Логин/пароль: Applanatest/Student2020!	Пользователь успешно авторизован, видит страницу «Панель инструментов».
2	Перейти в «Контрагенты» → «Контактные лица»	Пользователь находится на странице «Контактные лица», видит таблицу имеющихся контактных лиц, есть кнопка «Создать контактное лицо».
3	Нажать на кнопку «Создать контактное лицо»	Открыта страница создания контактного лица.
4	Заполнить обязательные поля: <ul style="list-style-type: none"><li>• фамилия;</li><li>• имя;</li><li>• организация;</li><li>• должность.</li></ul>	Поля заполнены.
5	Нажать на кнопку «Сохранить и закрыть»	Страница создания контактного лица закрыта, пользователь видит страницу «Все контактные лица» и всплывающее уведомление «Контактное лицо сохранено».