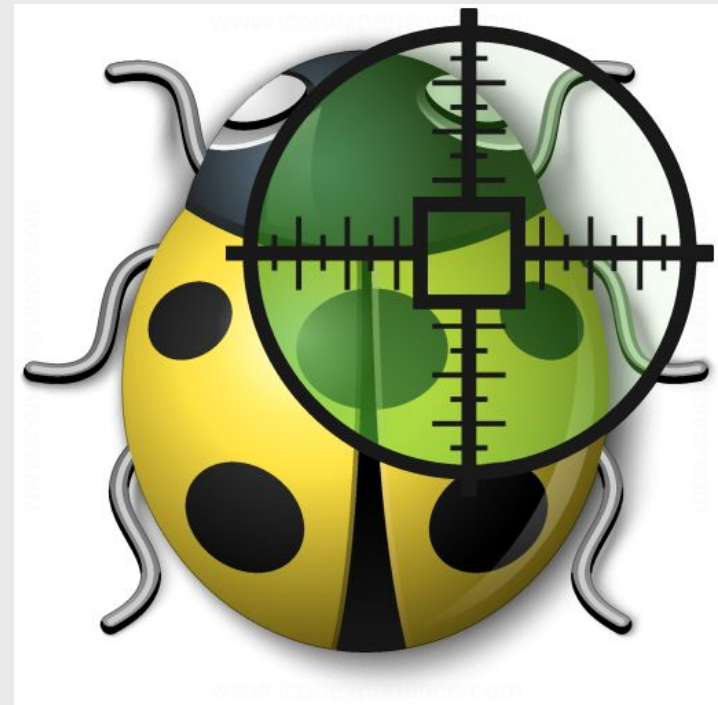
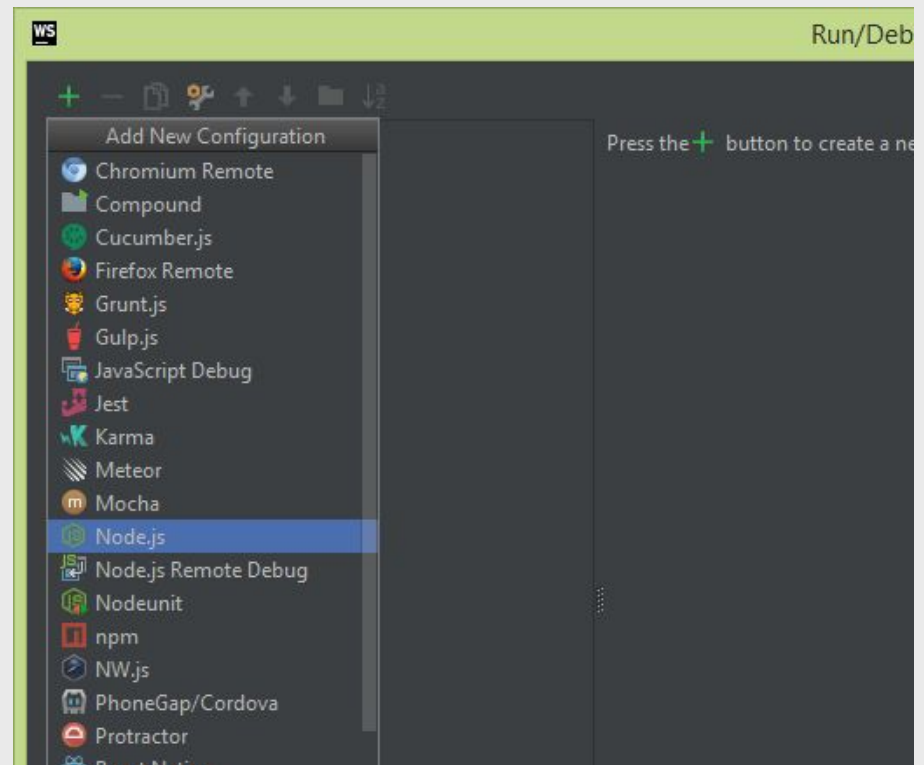
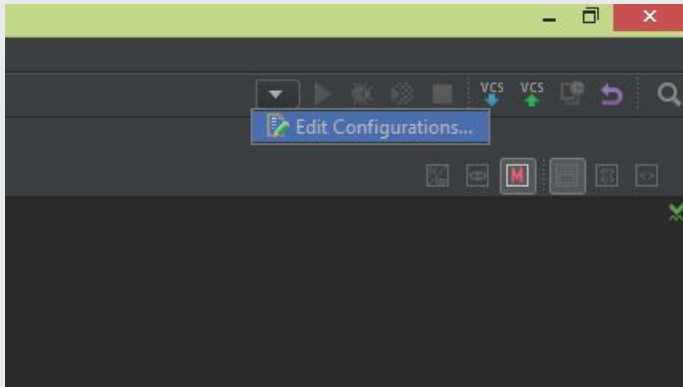


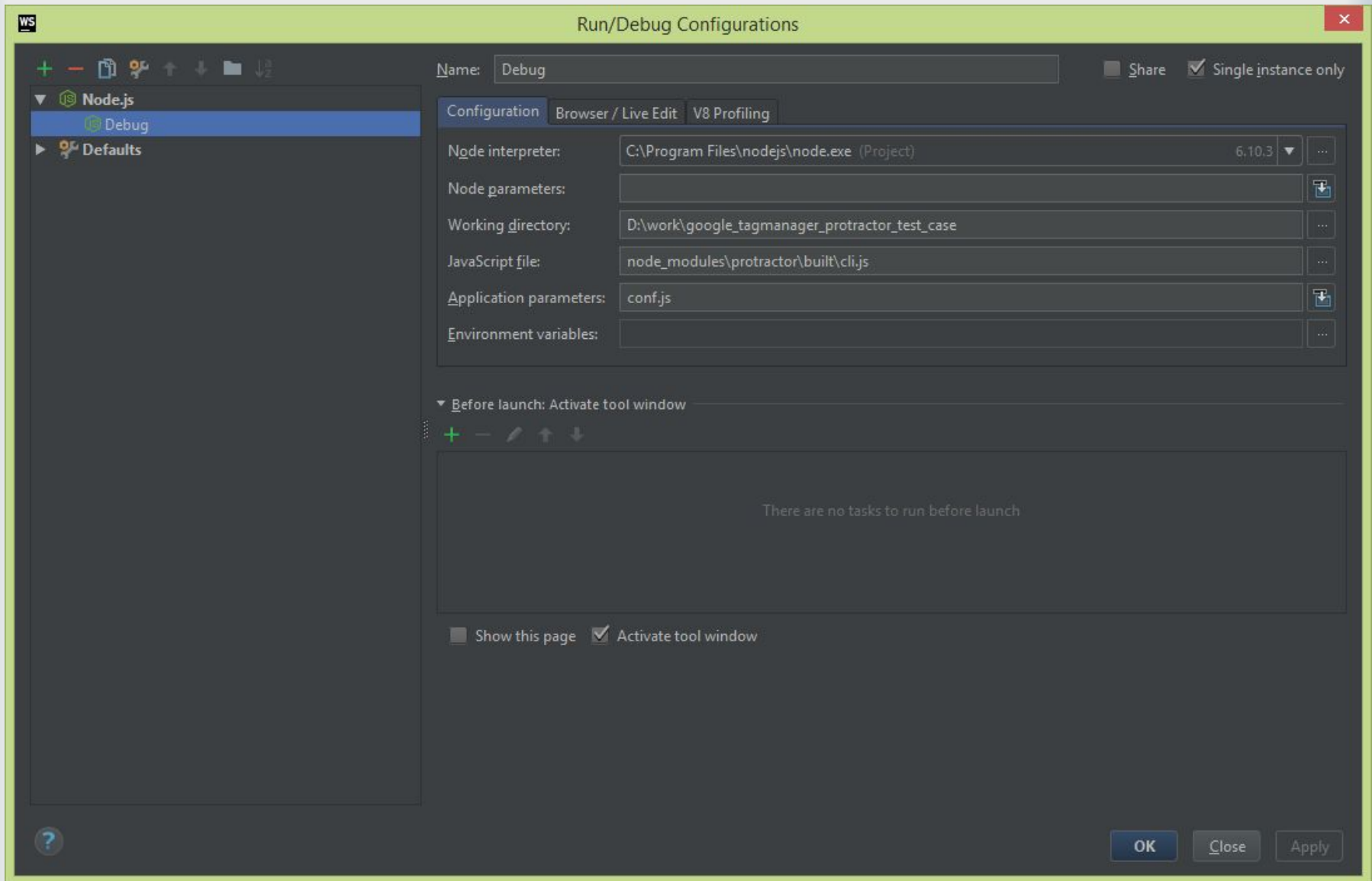
Debugging Protractor Tests in WebStorm



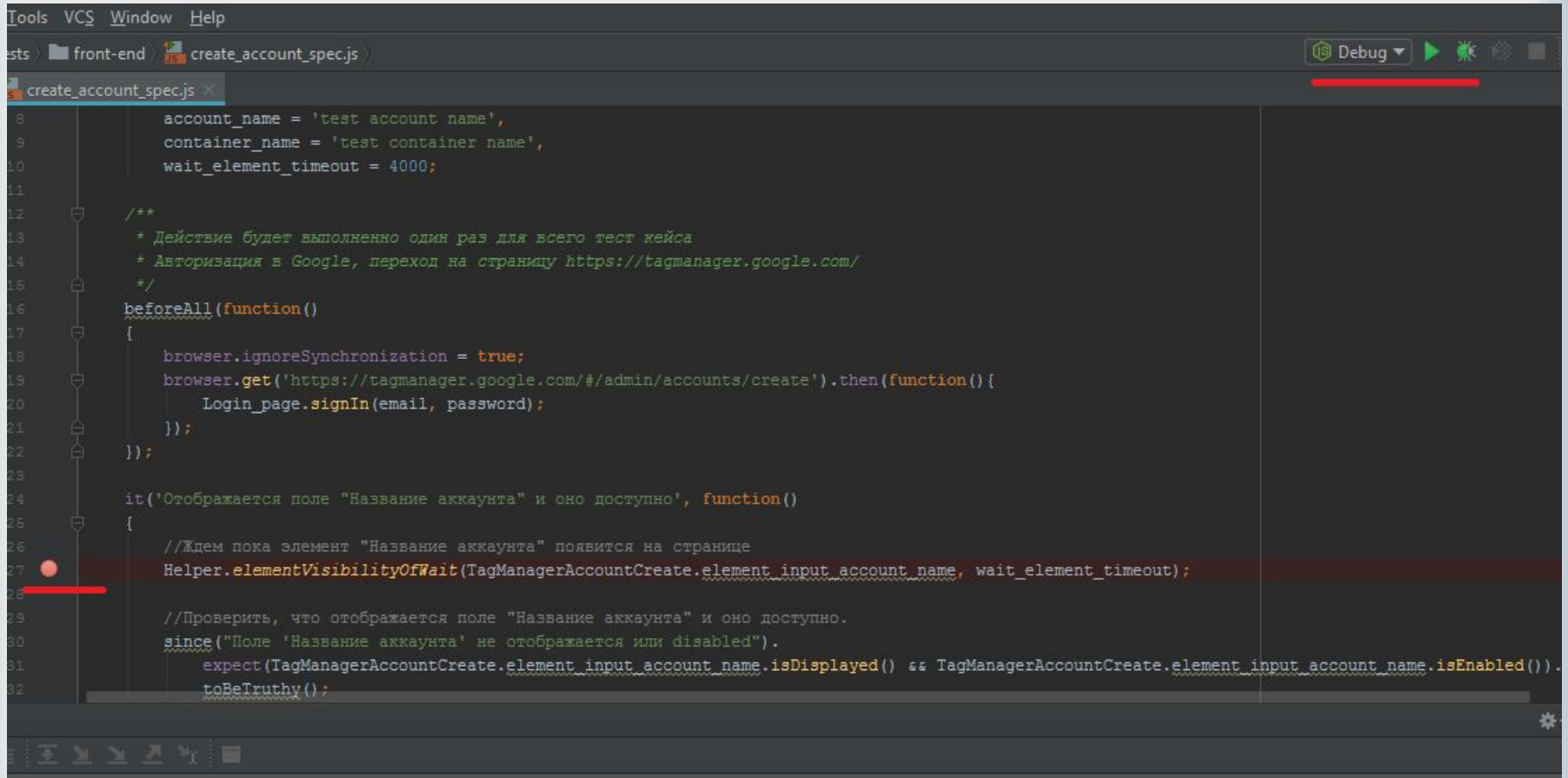
Настройка Build конфигурации



Настройка Build конфигурации



Настройка Build конфигурации



```
Tools VCS Window Help
ests > front-end > create_account_spec.js
create_account_spec.js x
8     account_name = 'test account name',
9     container_name = 'test container name',
10    wait_element_timeout = 4000;
11
12    /**
13     * Действие будет выполнено один раз для всего тест кейса
14     * Авторизация в Google, переход на страницу https://tagmanager.google.com/
15     */
16    beforeAll(function()
17    {
18        browser.ignoreSynchronization = true;
19        browser.get('https://tagmanager.google.com/#/admin/accounts/create').then(function() {
20            Login_page.signIn(email, password);
21        });
22    });
23
24    it('Отображается поле "Название аккаунта" и оно доступно', function()
25    {
26        //Ждем пока элемент "Название аккаунта" появится на странице
27        Helper.elementVisibilityOfWait(TagManagerAccountCreate.element_input_account_name, wait_element_timeout);
28
29        //Проверить, что отображается поле "Название аккаунта" и оно доступно.
30        since("Поле 'Название аккаунта' не отображается или disabled").
31            expect(TagManagerAccountCreate.element_input_account_name.isDisplayed() && TagManagerAccountCreate.element_input_account_name.isEnabled()).
32            toBeTruthy();
```

Настройка Build конфигурации

The image shows a code editor with a JavaScript file named `google_tag_manager_spec.js`. The code is a Jasmine test suite for a web application. A red circle highlights the local variables in the debug console, which are:

- `dvr = WebDriver`
- `browser = Browser`
- `browser.driver = WebDriver`
- `by = ProtractorBy`
- `LOGIN = [REDACTED]`
- `PASSWORD = [REDACTED]`
- `this = Object`

The code in the editor includes the following test functions:

```
afterEach(function () {
  var createButton = element(by.cssContainingText('.btn', 'Создать'));
  expect(createButton.isEnabled()).toBeFalsy();
});

it('should sign in and go to the page', function() {
  var dvr = browser.driver;
  dvr.get('https://tagmanager.google.com/#/admin/accounts/create');
  dvr.findElement(by.id('Email')).sendKeys(LOGIN);
  dvr.findElement(by.id('next')).click();
  browser.sleep(1000);
  dvr.findElement(by.id('Passwd')).click();
  dvr.findElement(by.id('Passwd')).sendKeys(PASSWORD);
  dvr.findElement(by.id('signIn')).click();
  browser.sleep(8000);
  expect(browser.getCurrentUrl()).toBe('https://tagmanager.google.com/#/adm');
});

it('should displayed and enabled text field "Name Account"', function() {
  var nameAccount = element(by.name('form.account.data.name'));
  expect(nameAccount.isDisplayed()).toBeTruthy();
  expect(nameAccount.isEnabled()).toBeTruthy();
});

it('should enter a value of name account', function() {
  var nameAccount = element(by.name('form.account.data.name'));
  nameAccount.sendKeys('Name Account');
  expect(nameAccount.getAttribute('value')).toBe('Name Account');
});
```

Отладка тестов Protractor

2 возможности отладки:

- `browser.pause()`
- `browser.debugger()`

browser.pause()

```
it('should fail to find a non-existent element', function() {  
  browser.get('app/index.html#/form');  
  browser.pause();  
  var nonExistant = element(by.binding('nopenopenope'))  
    .getText();  
});
```

После того, как тест остановился, можно использовать следующие команды:

- **c** для продвижения вперед
- **repl** для входа в интерактивный режим
- **ctrl-C** для выхода и продолжения теста

browser.debugger()

```
it('should fail to find a non-existent element', function() {  
  browser.get('app/index.html#/form');  
  browser.debugger();  
  var nonExistant = element(by.binding('nopenopenope'))  
    .getText();  
});
```

Для того, чтобы использовать отладчик, нужно стартовать Protractor с опцией отладки:

protractor debug conf.js