



**Don't hate the  
hate the salad** **cucumber,**

**Nikolaj Tolkačiov**





---

# Gherkin background





→ **Protractor era**

→ **Specflow era**

→ **Calabash era**



*Calaba.sh*



**era**





**era**





**Protractor**  
end to end testing for AngularJS



**era**





# Fully Realized Gherkin Project



# **Maximize** reusability

Keep step definition count to minimum to increase reusability.

- Feature step writing became easier as there is fewer step definitions you can choose from
- Lower maintenance
- Step definitions became more flexible



# Reusability statistics

Did that decision payed-off over 3 years of frameworks life cycle?

- Total Step count: **12579 (+1811 including examples)**
- Total step definition count: **53**
- Average: **237(271)** steps/step definition
- Median ~ **500** steps/step definition



---

# Step Definitions



# Step Definitions

```
Then(/^I (?:(click|press) (?:(on |))(.*) (button|checkbox|tab|card|row|hyperlink|modal button|accordion)$/, async function (buttonName, buttonType) {  
    return await ClickableFunctions.clickOnButton(`${buttonName} ${buttonType}`)  
})
```

```
Then(/^I (?:should|can) see (.*) (as|starting with|containing|more than) "([^"]*)"$/, async function (fieldName, matchType, fieldValue) {  
    return await FieldFunctions.shouldSeeFieldWithValue(fieldName, fieldValue, matchType)  
});
```



# Step Functions

```
clickOnButton: async function (buttonName) {  
  const button = await ElementParser.getClickable(buttonName)  
  return assert(await ElementInteractions.clickIfExists(button), `${buttonName} not found`)  
}
```

```
shouldSeeInDropDown: async function (count, value, dropDownName) {  
  const dropDown = await ElementParser.getDropDown(dropDownName)  
  const foundCount = await dropDown.all(by.cssContainingText('option', value)).count()  
  return assert(foundCount === count, `Expected to see ${count} of ${value}, but found ${foundCount}`)  
}
```



# Error message

**FAIL** src/App.spec.js

- App > should render the App Component

```
expect(received).toEqual(expected)
```

Expected value to equal:

true

Received:

false



# Step functions

```
clickOnButton: async function (buttonName) {  
  const button = await ElementParser.getClickable(buttonName)  
  return assert(await ElementInteractions.clickIfExists(button), `${buttonName} not found`)  
}
```

```
shouldSeeInDropDown: async function (count, value, dropDownName) {  
  const dropDown = await ElementParser.getDropDown(dropDownName)  
  const foundCount = await dropDown.all(by.cssContainingText('option', value)).count()  
  return assert(foundCount === count, `Expected to see ${count} of ${value}, but found ${foundCount}`)  
}
```



1000



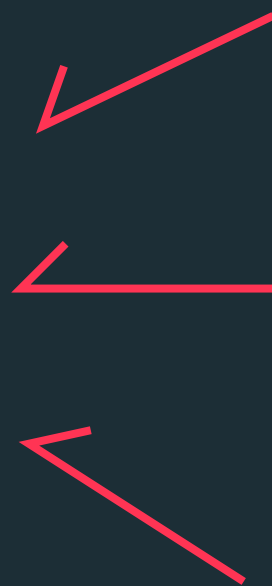
# Vocabulary

```
module.exports = async function (dropDownName) {
  switch (Strings.sanitizeString(dropDownName)) {
    case "room name":
      return Locators.ordersPage.orderManageRoomName
    case "credit order status":
      return Locators.ordersPage.creditOrderStatusDropDown
    case "room inventory":
      return Locators.ordersPage.orderManageRoomInventory
    case "order room status":
      return Locators.ordersPage.orderManageRoomOrderRoomStatus
    case "event search by":
    case "venue search by":
      return Locators.submitForms.searchByDropDown
    default:
      const errorText = ` '${dropDownName}' is not defined yet.`
      if (Strings.sanitizeString(dropDownName) !== dropDownName) {
        errorText += ` Post sanitize value: ${Strings.sanitizeString(dropDownName)} `
      }
      return assert(false, errorText)
  }
}
```



# Vocabulary

FIELDS



isFieldVisible  
(...)

getFieldValue  
(...)

enterValueToField  
(...)





---

# Page Detection



# Page Detection

```
case "status":
    switch (await browser.getTitle()) {
        case Locators.assetsPage.assetSearchPageTitleText:
            return Locators.assetsPage.statusDropDown
        case Locators.assetDetailsPage.assetCreationPageTitleText:
        case Locators.assetDetailsPage.assetEditPageTitleText:
            return Locators.assetDetailsPage.currentStatus
        default:
            return assert(false, "'" + await browser.getTitle() + "' page is not defined yet")
    }
```



---

# Injecting Locators



# Injecting locators

```
eventEventName: element(by.className(`e2e-event-name`)),  
eventStartDate: element(by.className(`e2e-event-start-date`)),  
eventEndDate: element(by.className(`e2e-event-end-date`)),
```

→ **View**



→ **Edit**





---

**Placeholder for  
ordinal numbers**



# Ordinal numbers

```
module.exports = function(pageName) {  
  switch (Strings.sanitizeString(pageName)) {  
    case "login":  
      return Locators.loginPage.get()  
    case "dashboard":  
      return Locators.dashboardPage.get()  
    case "[anyOrdinalNumber] event overview":  
      return Locators.eventsPage.getTestEventOverviewPage(Strings.getIndexFromString(pageName))  
    case "[anyOrdinalNumber] order overview":  
      return Locators.ordersPage.getTestOrderOverviewPage(Strings.getIndexFromString(pageName))  
  }  
}
```



# Ordinal numbers

```
testEventIds: [  
  { number: '1', id: 13 },  
  { number: '2', id: 2313 },  
  { number: '3', id: 230 },  
  { number: '4', id: 41260 },  
  { number: '5', id: 41282 },  
  { number: '6', id: 41291 },  
  { number: '7', id: 41294 },  
  { number: '8', id: 41295 }  
],
```



# Ordinal numbers

```
switch (Strings.sanitizeString(rowName)) {  
    case "[anyOrdinalNumber] tax exemption":  
        return Strings.appendIndexFromStringOrReturnFirst(Locators.locationPage.taxExemptionList, rowName)  
    case "[anyOrdinalNumber] asset item":  
        return Strings.appendIndexFromStringOrReturnFirst(Locators.assetsPage.assetItemsList, rowName)  
}
```





Page / Component  
objects



# Page/component objects

```
const { element, by } = require('protractor')

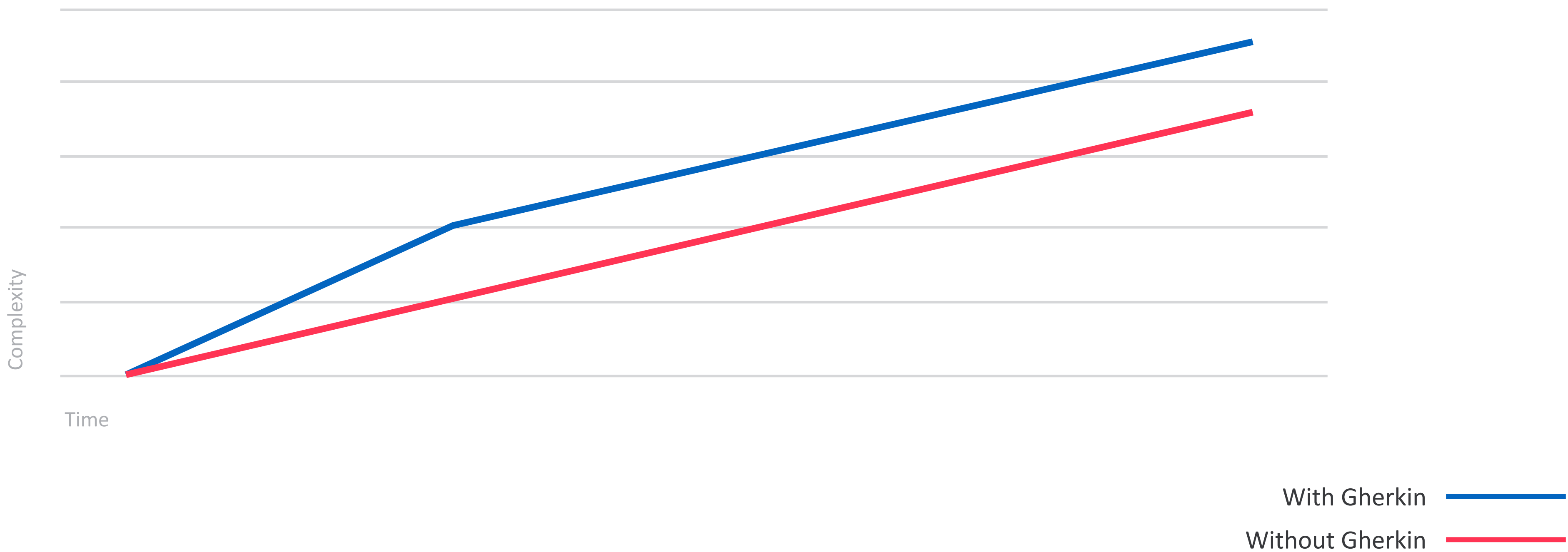
let formObject = {
  //DOM elements:
  saveButton: element(by.className('e2e-form-save-button')),
  cancelButton: element(by.className('e2e-form-cancel-button')),
  formBackButton: element(by.className('e2e-form-back-button')),
  querySearchButton: element(by.className('e2e-search-button')),
  flyoutLunchEditModeButton: element(by.className('e2e-flyout-lunch-edit')),
  flyoutPreviousButton: element(by.className('e2e-flyout-previous')),
  flyoutNextButton: element(by.className('e2e-flyout-next')),
  orderRangeFilterField: element(by.className('e2e-datepicker-range-field')),
  resetFiltersButton: element(by.className('e2e-reset-filters')),
  venueNameInput: element(by.className('e2e-venue-name-input')),
  vendorSelectField: element(by.className('e2e-vendor-select-field')),
  overviewNotesCard: element(by.className('e2e-notes-card')),
  overviewMostRecentNoteField: element(by.className('e2e-notes-card-recent')),
  overviewNotesCardCountField: element(by.className('e2e-notes-card-count')),
}

module.exports = formObject
```



# Summary

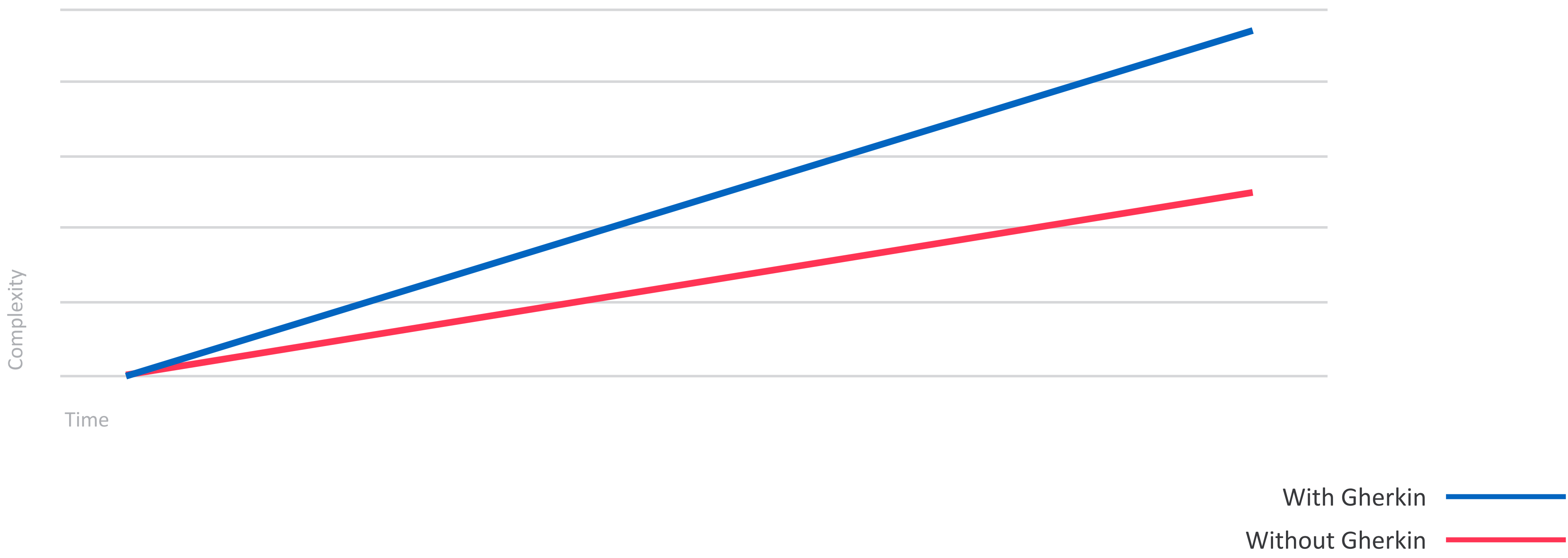
Gherkin **VS** non-Gherkin





# Summary

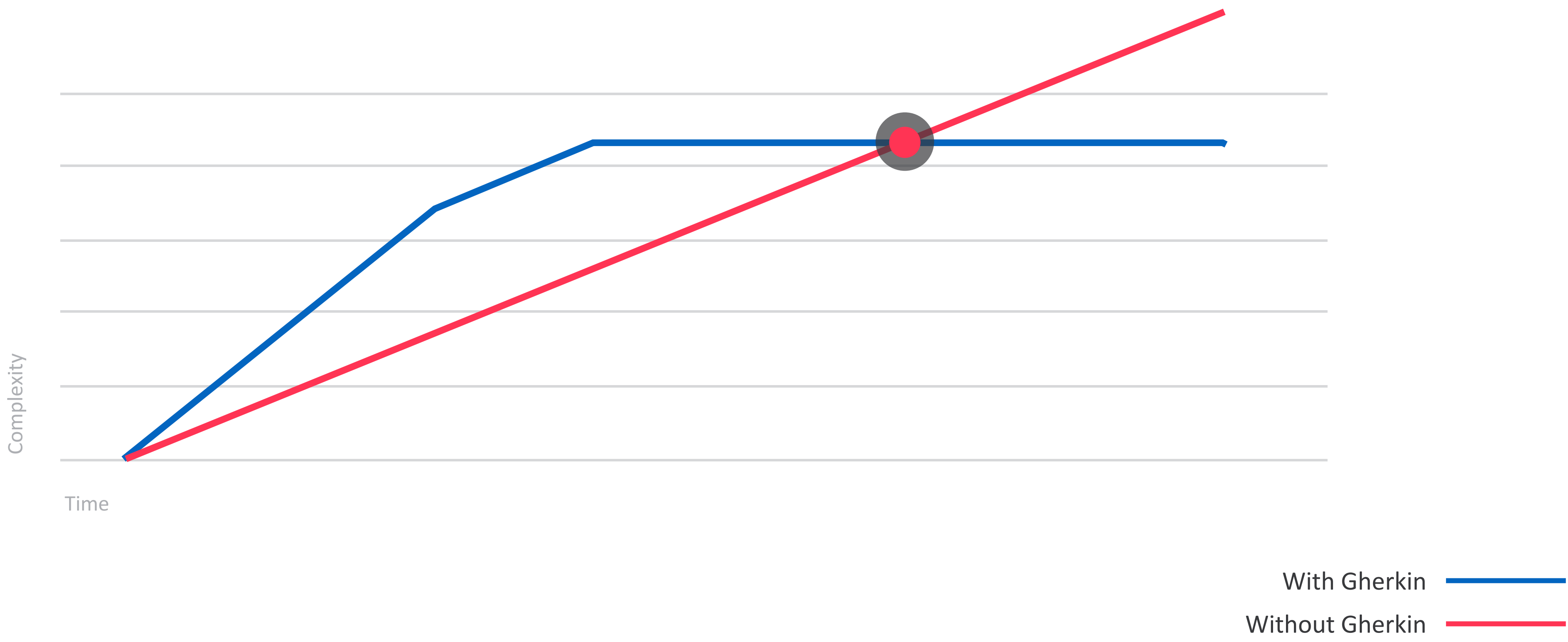
Gherkin **VS** non-Gherkin












# Summary

Gherkin **VS** non-Gherkin





# Summary

- 1**  Maximize reusability
- 2**  Informative error messages
- 3**  Vocabulary for mapping text to element
- 4**  Page detection to minimize text overhead for variables
- 5**  Injecting locators to minimize text overhead for variables
- 6**  Ordinal number placeholder to decrease vocabulary size
- 7**  Page/component objects to minimize duplicates





# Thank you.

**Nikolaj Tolkačiov**



Nikolaj.Tolkaciov@devbridge.com



Nikolaj Tolkaciov