OK.TECH

Как заставить API самому себе писать тесты:

практика генерации тестов на основе спецификации API

Елизавета Андреева и Николай Борисенко

Спикер

Инженер по автоматизации тестирования.

Занимаюсь автоматизацией под Web, Mobile web, API и Android платформы.

Обучаю автоматизации коллег и студентов в VK Образовании.

Елизавета Андреева

Спикер

Fullstack — тестировщик в команде антиспам.

Занимаюсь автоматизацией под Web, Mobile web, API, Android и iOS платформы.

Обучаю автоматизации коллег.

Николай Борисенко

План

→ Автотесты в ОК

План

- Автотесты в OK
- Наше API

План

- Автотесты в OK
- Наше API
- Куда мы генерируем тесты

План

- Автотесты в OK
- Наше API
- Куда мы генерируем тесты
- Процесс генерации в CI/CD

План

- Автотесты в OK
- Наше API
- Куда мы генерируем тесты
- Процесс генерации в CI/CD
- Шаблоны тестов

План

- Автотесты в OK
- Наше API
- Куда мы генерируем тесты
- Процесс генерации в CI/CD
- Шаблоны тестов
- Реализация генератора

План

- Автотесты в OK
- Наше API
- Куда мы генерируем тесты
- Процесс генерации в CI/CD
- Шаблоны тестов
- Реализация генератора
- Шаги генерации

План

- Автотесты в OK
- Наше API
- Куда мы генерируем тесты
- Процесс генерации в CI/CD
- Шаблоны тестов
- Реализация генератора
- Шаги генерации
- Результаты

План

- Автотесты в OK
- Наше API
- Куда мы генерируем тесты
- Процесс генерации в CI/CD
- Шаблоны тестов
- Реализация генератора
- Шаги генерации
- Результаты
- Планы

OK.TECH

Платформа	API	Web	Mobile	iOS	Android
Количество тестовых классов	~3600	~3200	~1200	~1100	~1400

OK.TECH

API — самая популярная платформа

Проблема

Ресурсов не хватает

Проблема

Ресурсов не хватает

Большой ручной регресс

Проблема

Ресурсов не хватает

Большой ручной регресс

Монотонная работа

Решение

Автогенерация тестов

Шаблонные проверки покрыты

Решение

Автогенерация тестов

Шаблонные проверки покрыты

Больше времени на сложные ручные кейсы

Решение

Автогенерация тестов

Шаблонные проверки покрыты

Больше времени на сложные ручные кейсы

Покрытие только кейсов на бизнес логику

Ключевые компоненты

Плюсы нашего клиента

→ Код понятен и разработчику, и тестировщику

Плюсы нашего клиента

- Код понятен и разработчику, и тестировщику
- Используем на всех платформах

Плюсы нашего клиента

- Код понятен и разработчику, и тестировщику
- Используем на всех платформах
- Специальные аннотации

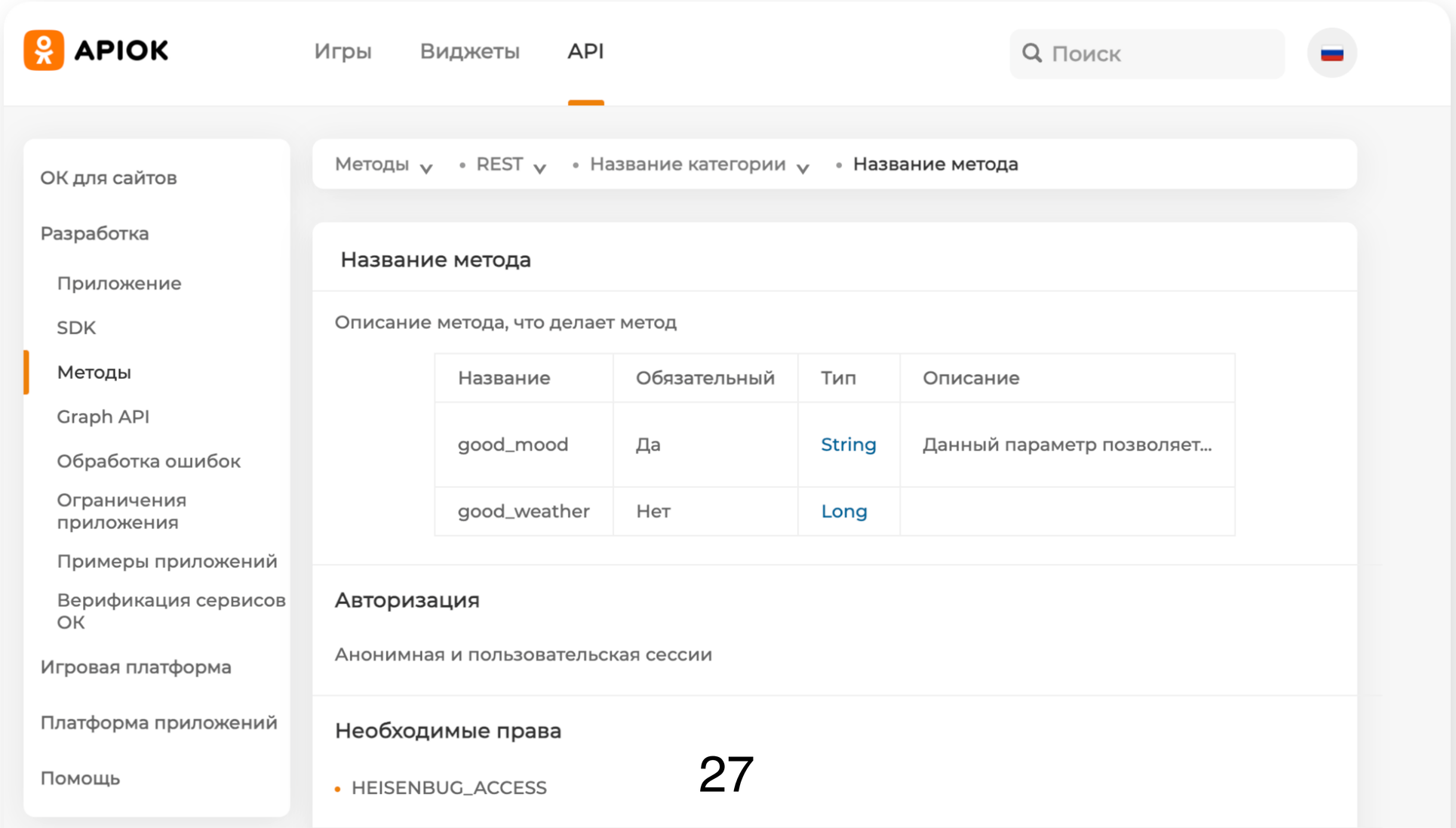
Плюсы нашего клиента

- Код понятен и разработчику, и тестировщику
- Используем на всех платформах
- Специальные аннотации
  - Автоматизация задач

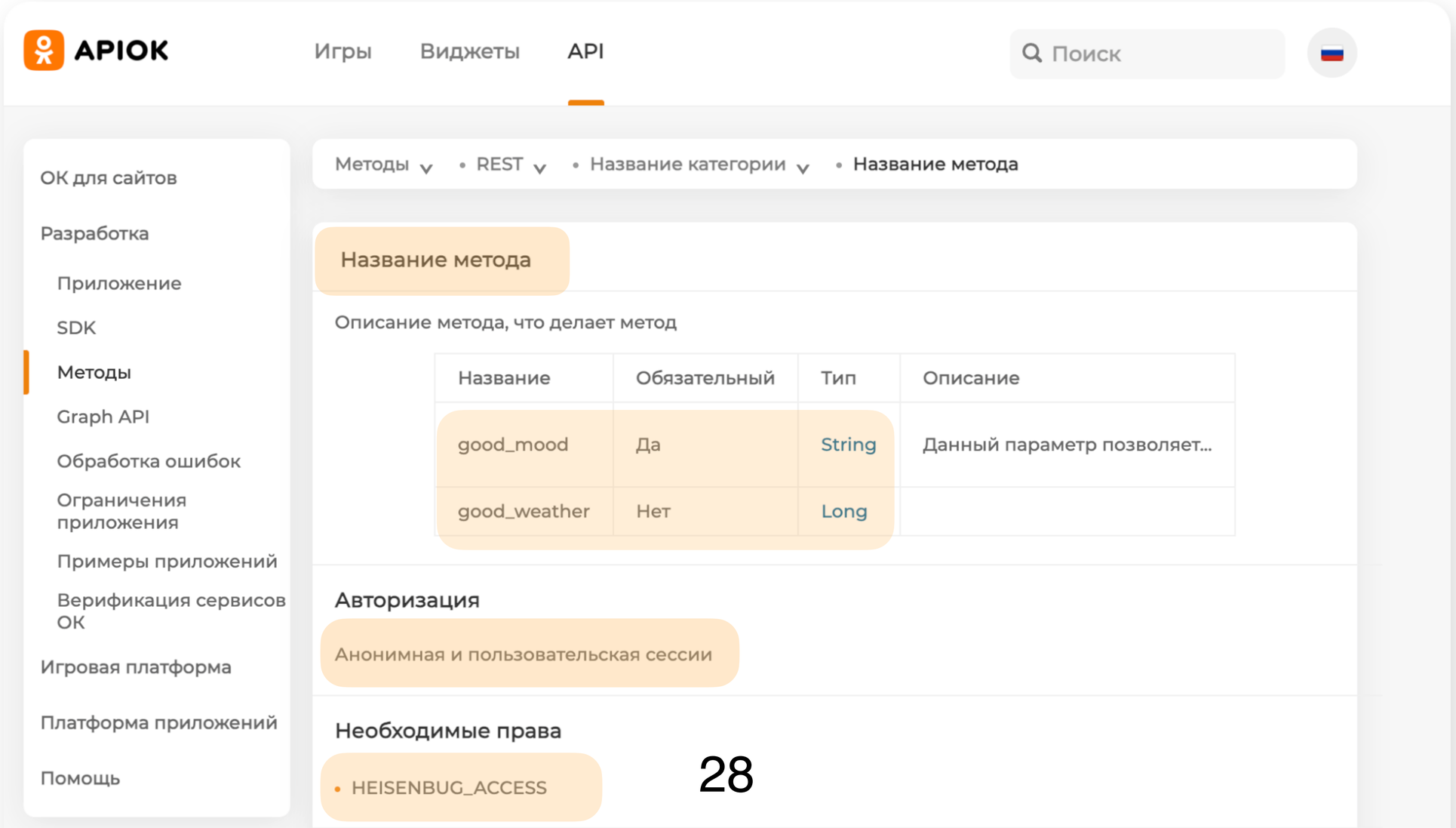
Плюсы нашего клиента

- Код понятен и разработчику, и тестировщику
- Используем на всех платформах
- Специальные аннотации
  - Автоматизация задач
  - Генерация документации

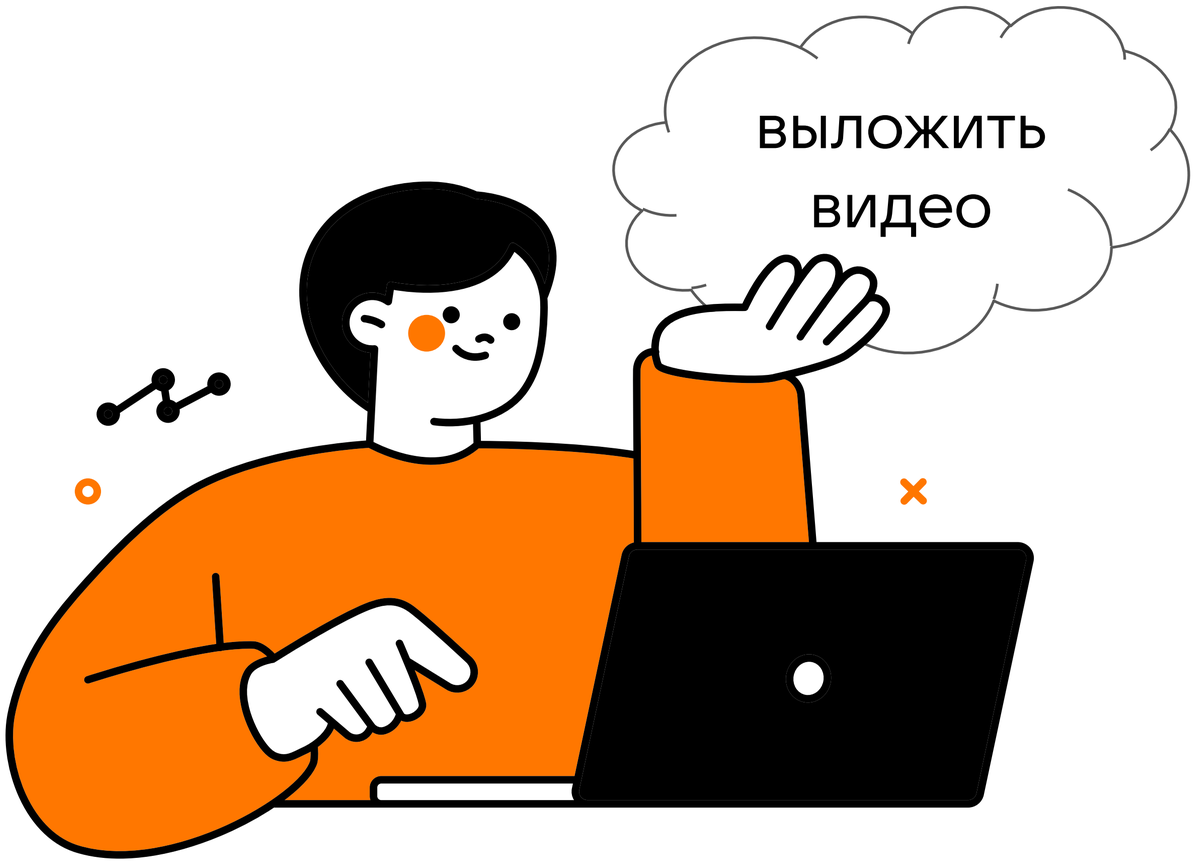
Генерация документации



Генерация документации



API методы



API методы

Метод, чтобы выложить наш доклад в ОК

Пользовательская сессия

Как добавить API метод?

- → Endpoint
- → Interface + Implementation
- → Response
- ✳ Data types

API метод

```
public interface HeisenbugService {


UploadHeisenbugVideoResponse uploadHeisenbugVideo(
```

Response API метода

```
public class UploadHeisenbugVideoResponse {


public boolean success;
```

Response API метода

```
public class UploadHeisenbugVideoResponse {


public boolean success;

public boolean isSuccess() {
    return success;
}
```

API метод

```
public interface HeisenbugService {


UploadHeisenbugVideoResponse uploadHeisenbugVideo(
```

API метод

```
public interface HeisenbugService {


@ApiMethodEndpoint(ApiMethod.UPLOAD_HEISENBUG_VIDEO)
UploadHeisenbugVideoResponse uploadHeisenbugVideo(
```

Endpoint API метода

```
UPLOAD_HEISENBUG_VIDEO(
(short)1,
"heisenbug",
"uploadHeisenbugVideo",
ApiProtectionLevel.SESSION_REQUIRED,
new Flags[]{Flags.PUBLIC}
),
```

id

категория

название метода

тип сессии

доп флаги

API метод

```
public interface HeisenbugService {


@ApiMethodEndpoint(ApiMethod.UPLOAD_HEISENBUG_VIDEO)
UploadHeisenbugVideoResponse uploadHeisenbugVideo(
```

API метод

```
public interface HeisenbugService {


@ApiMethodEndpoint(ApiMethod.UPLOAD_HEISENBUG_VIDEO)
UploadHeisenbugVideoResponse uploadHeisenbugVideo(
@RestParam(value = "name", required = true) String var1,
@RestParam(value = "pathToFile", required = true) File var2),
@RestParam("info") VideoInfo var3));
```

Аннотация RestParam

```java
@Retention(RetentionPolicy.RUNTIME)
@Target({ElementType.PARAMETER})
public @interface RestParam {
    String value();

    boolean required() default false;
}
```

Аннотация RestParam

```java
@Retention(RetentionPolicy.RUNTIME)
@Target({ElementType.PARAMETER})
public @interface RestParam {
    String value();

    boolean required() default false;
}
```

```java
@RestParam(
value = "name",
required = true)
String var
```

Как добавить API метод?

- ~~Endpoint~~
- ~~Interface + Implementation~~
- ~~Response~~

API методы

API клиент

```
public interface ApiClient {

    HeisenbugService getHeisenbugService();

    AnotherService getAnotherService();

                              ...
}
```

API методы

```
apiClient
```

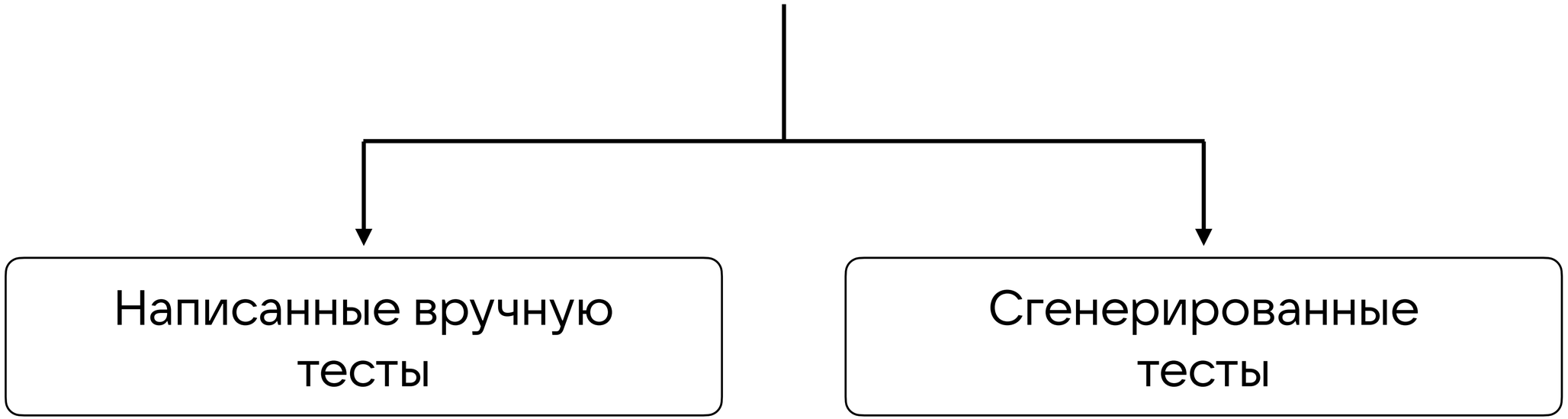
API методы

```
apiClient
  .getHeisenbugService()
```

API методы

```
apiClient
  .getHeisenbugService()
  .uploadHeisenbugVideo(
    "API генератор",
    new File("/HB"),
    new VideoInfo("desc", true))
```

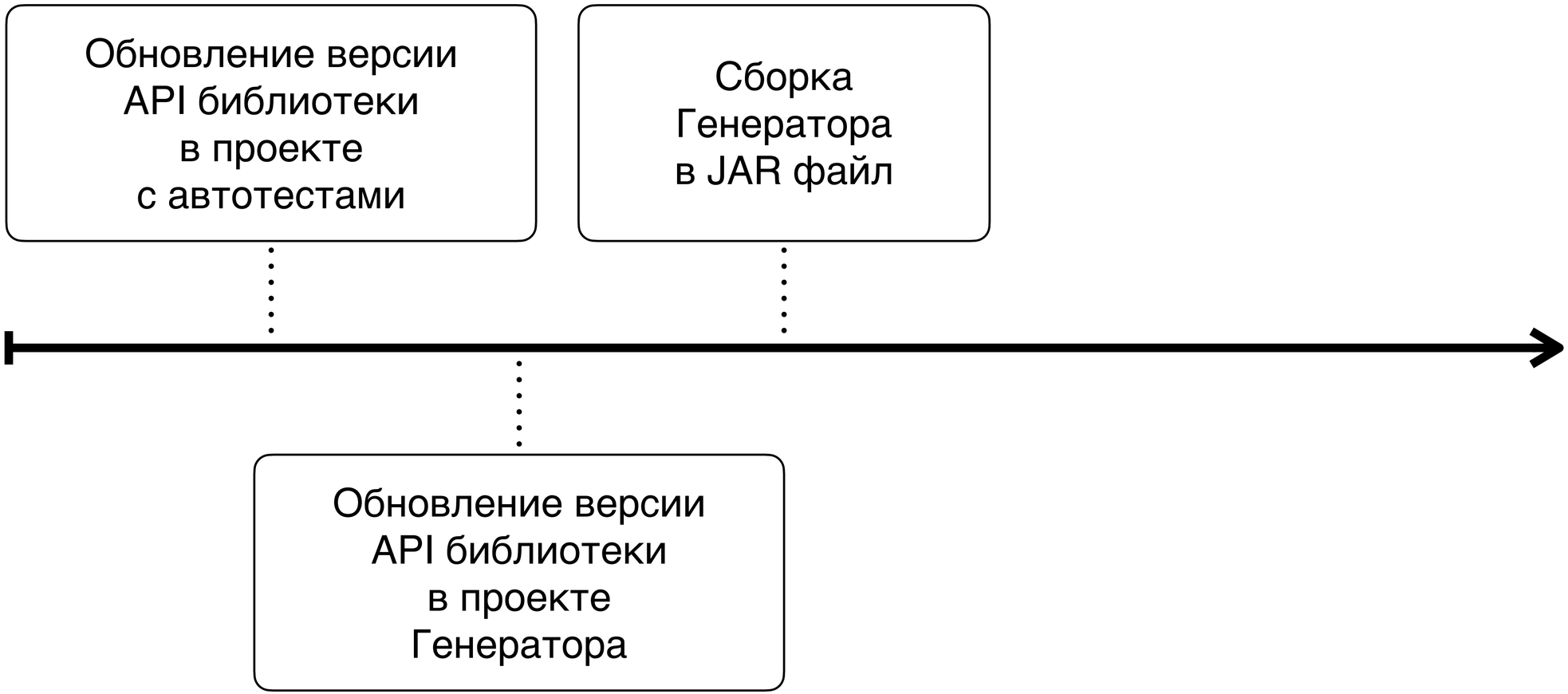
Проект API тестов



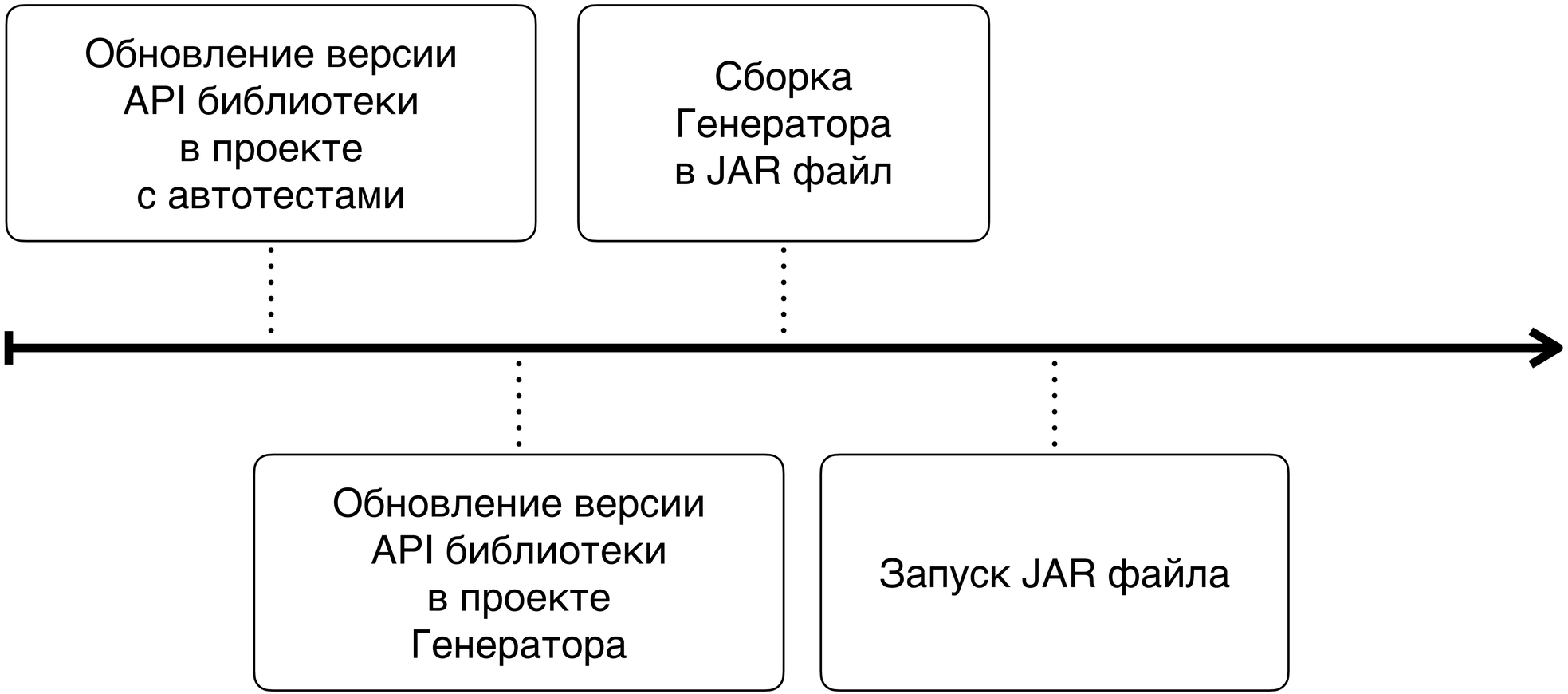
Шаги генерации в TeamCity

Шаги генерации в TeamCity

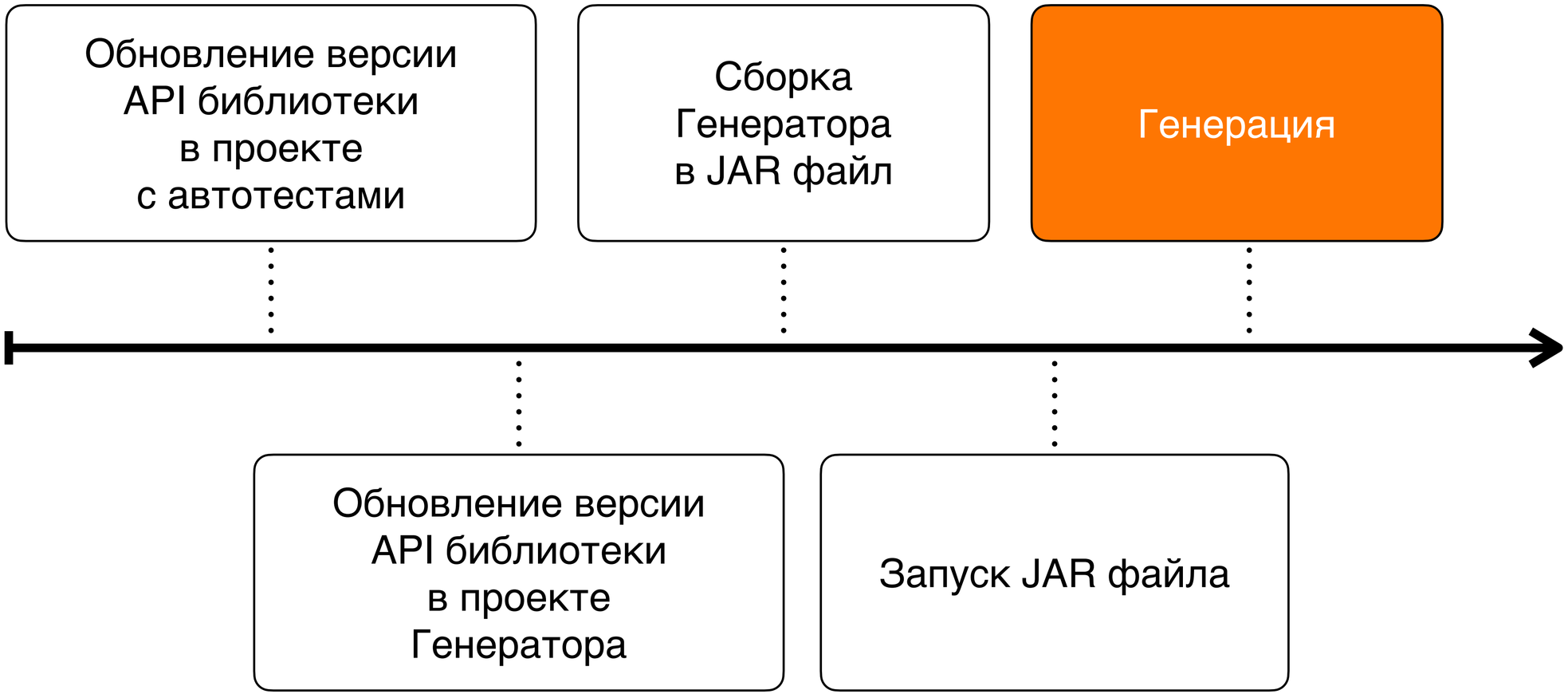
Шаги генерации в TeamCity



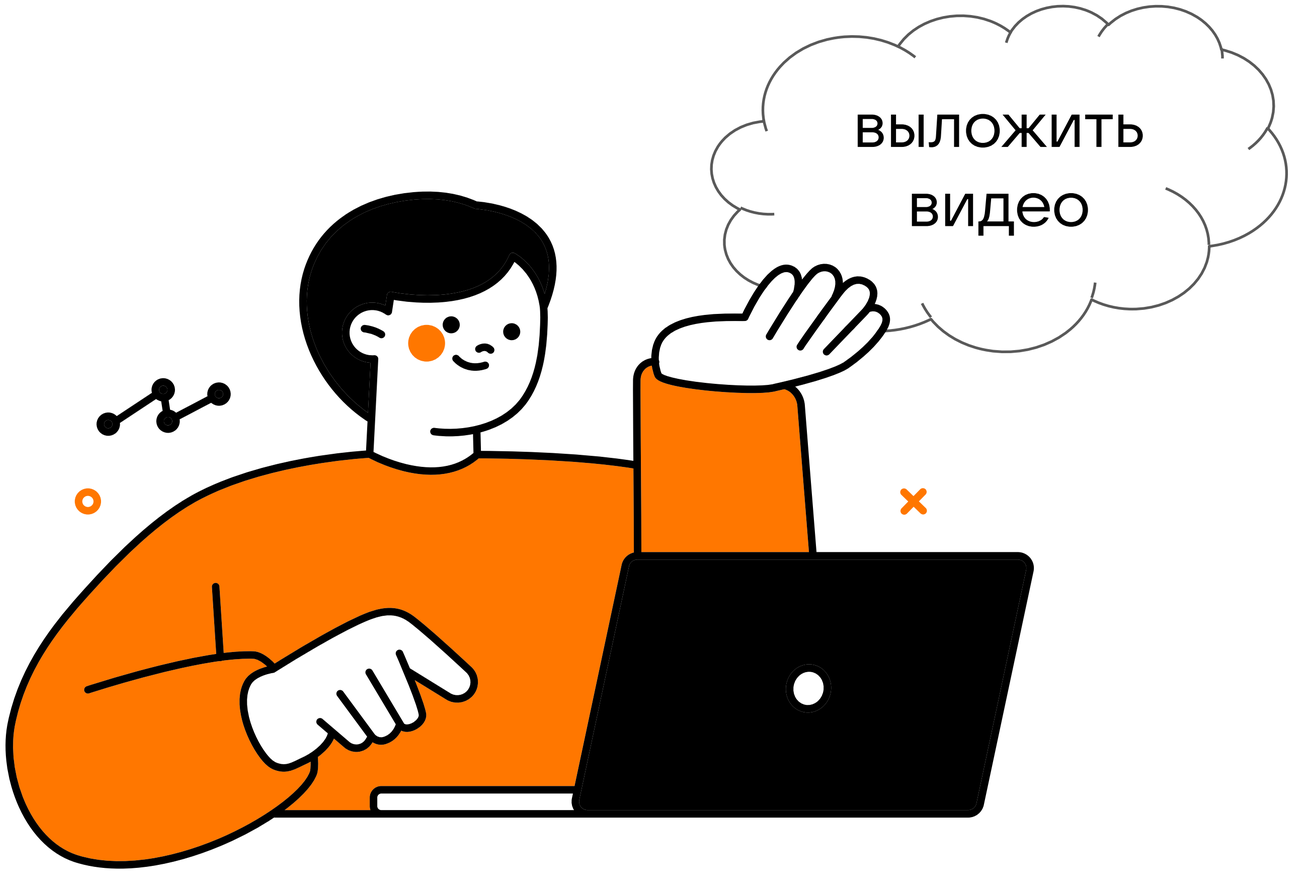
Шаги генерации в TeamCity



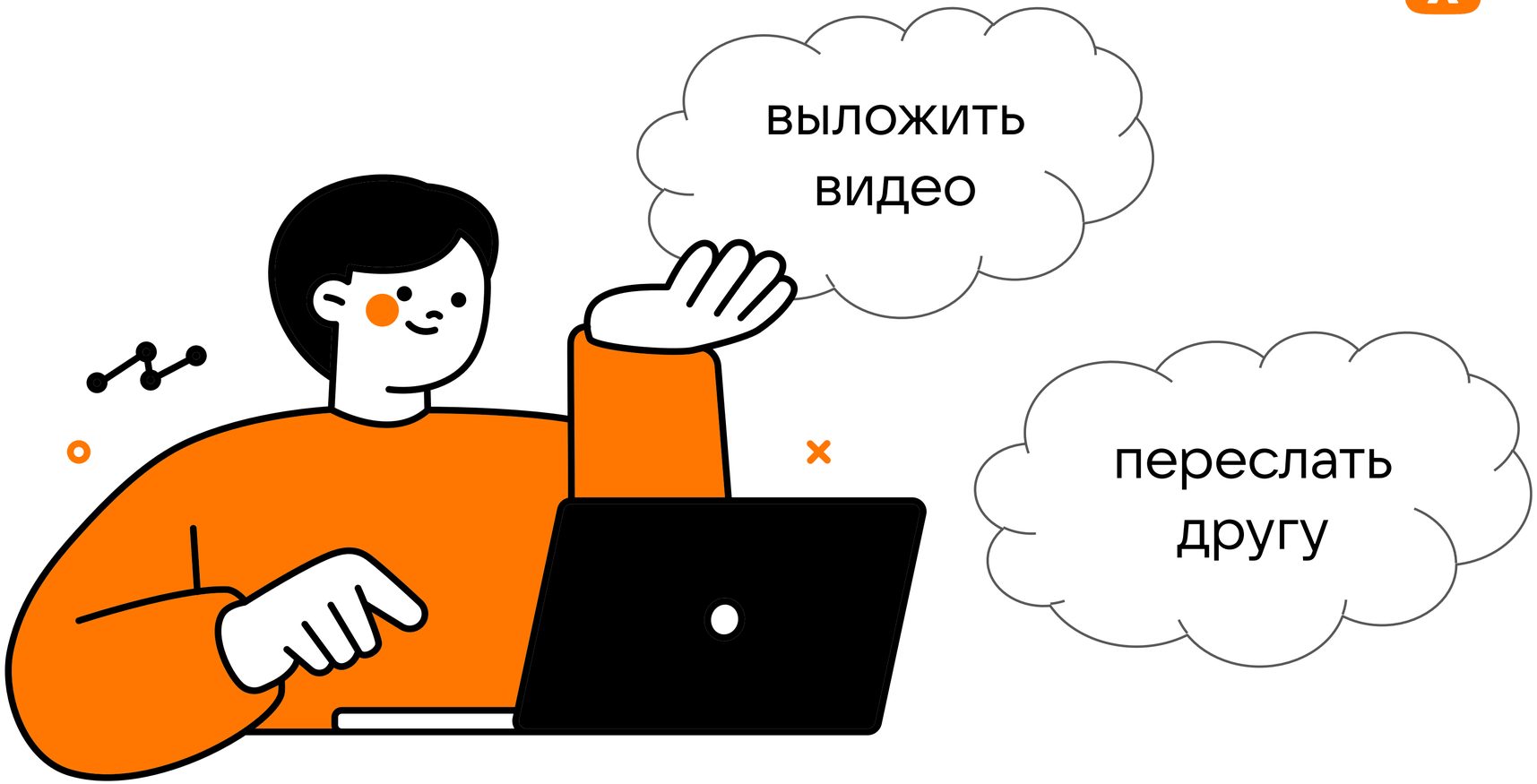
Шаги генерации в TeamCity



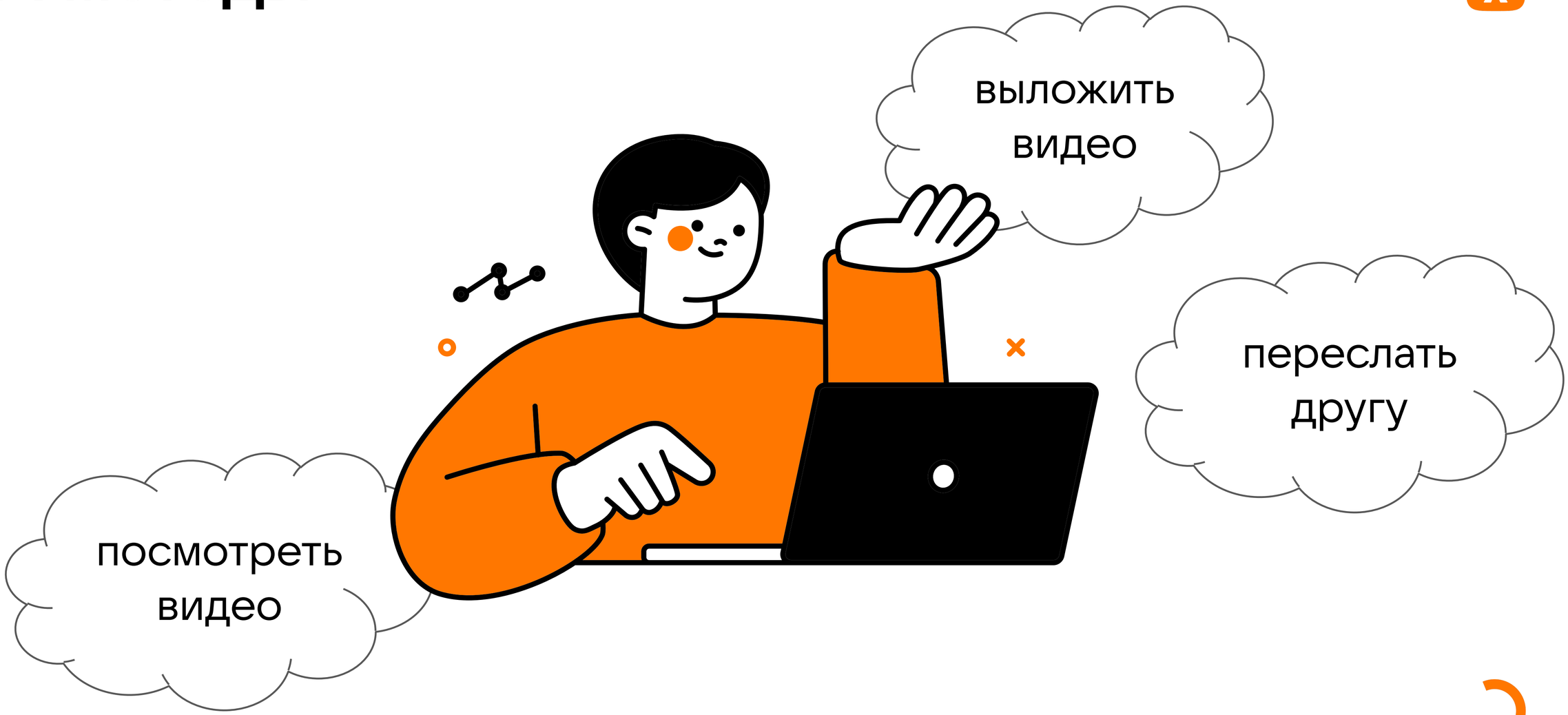
API методы



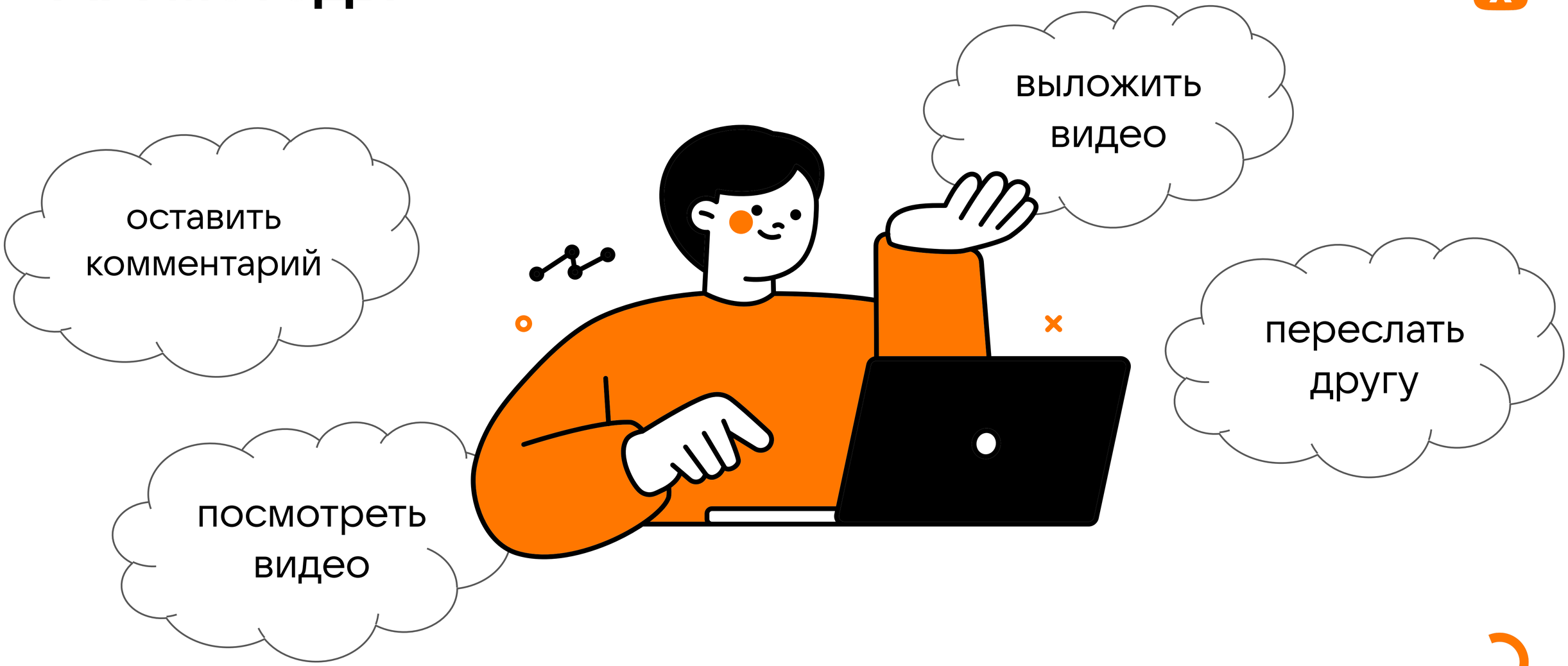
API методы



API методы



API методы

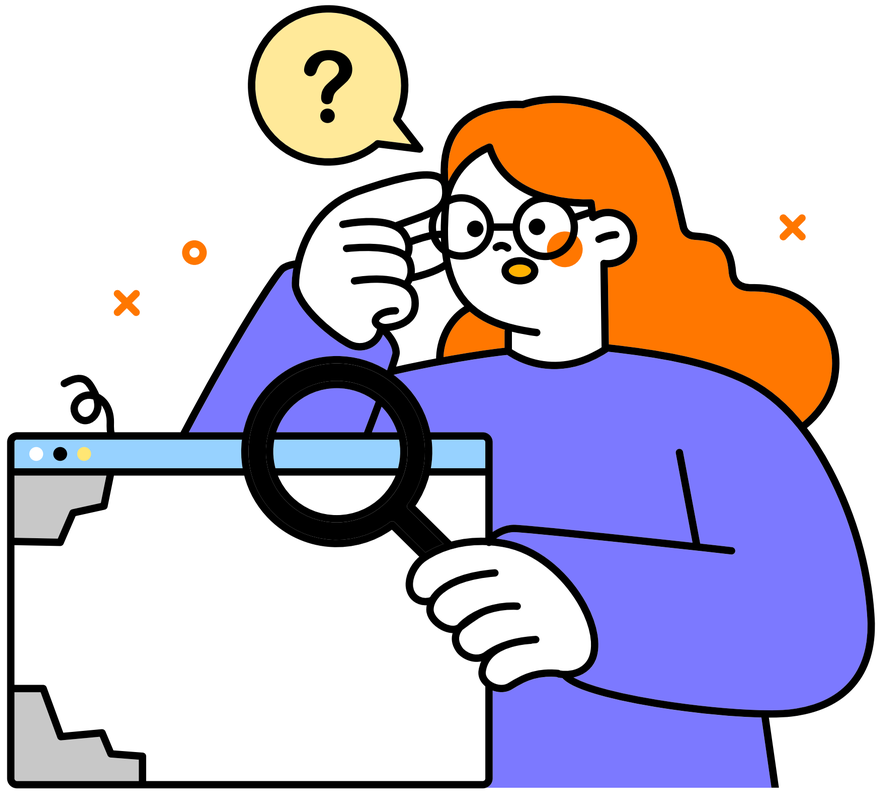

выложить видео
оставить комментарий
переслать другу
посмотреть видео

Задача
на автоматизацию
API метода —
Оставить комментарий

Задача
на автоматизацию
API метода —
Выложить видео

Задача
на автоматизацию
API метода —
Посмотреть видео

Задача
на автоматизацию
API метода —
Переслать другу



Список методов

Комментарий к видео	**Загрузка видео**
Параметры: 1) Видео ID — обязательный параметр 2) Текст — обязательный параметр	**Параметры:** 1) Название - необязательный параметр 2) Путь к файлу - обязательный параметр 3) Дополнительная информация — необязательный параметр
Тип сессии: Анонимная сессия	**Тип сессии:** Пользовательская сессия
Просмотр видео	**Отправка другу видео**
Параметры: 1) Видео ID — обязательный параметр	**Параметры:** 1) ID друга — обязательный параметр 2) ID видео — обязательный параметр 3) Текст — необязательный параметр
Тип сессии: Без сессии	**Тип сессии:** Пользовательская сессия

Комментарий к видео	**Загрузка видео**
Параметры: 1) Видео ID — обязательный параметр 2) Текст — обязательный параметр	**Параметры:** 1) Название — необязательный параметр 2) Путь к файлу — обязательный параметр 3) Дополнительная информация — необязательный параметр
Тип сессии: Анонимная сессия	**Тип сессии:** Пользовательская сессия

Блок названия

Комментарий к видео	Загрузка видео
Параметры: 1) Видео ID — обязательный параметр 2) Текст — обязательный параметр	**Параметры:** 1) Название — необязательный параметр 2) Путь к файлу — обязательный параметр 3) Дополнительная информация — необязательный параметр
Тип сессии: Анонимная сессия	**Тип сессии:** Пользовательская сессия

Блок параметров

Комментарий к видео	Загрузка видео
Параметры: **1) Видео ID** — обязательный параметр **2) Текст** — обязательный параметр	**Параметры:** **1) Название** — необязательный параметр **2) Путь к файлу** — обязательный параметр **3) Дополнительная информация** — необязательный параметр
Тип сессии: Анонимная сессия	**Тип сессии:** Пользовательская сессия

Блок параметров

Комментарий к видео	Загрузка видео
Параметры: 1) Видео ID — **обязательный параметр** 2) Текст — **обязательный параметр**	**Параметры:** 1) Название — **необязательный параметр** 2) Путь к файлу — **обязательный параметр** 3) Дополнительная информация — **необязательный параметр**
Тип сессии: Анонимная сессия	**Тип сессии:** Пользовательская сессия

Блок типа сессии

Комментарий к видео	Загрузка видео
Параметры: 1) Видео ID — обязательный параметр 2) Текст — обязательный параметр	**Параметры:** 1) Название — необязательный параметр 2) Путь к файлу — обязательный параметр 3) Дополнительная информация — необязательный параметр
Тип сессии: **Анонимная сессия**	**Тип сессии:** **Пользовательская сессия**

Какие тесты мы можем генерировать?

- Вызовы методов в некорректных сессиях
- Вызовы метода без одного обязательного параметра
- Вызовы метода без нескольких обязательных параметров

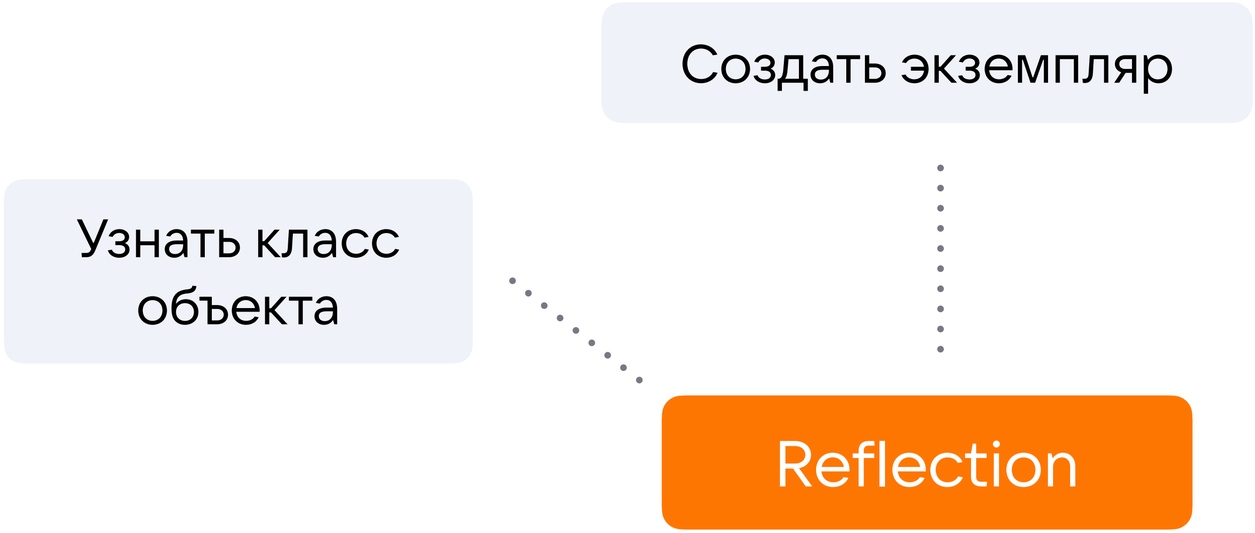
Языки программирования с Reflection	Java, Python, Kotlin, Go …
API клиент	Любой API клиент написанный на языках, у которых есть механизм Reflection
CI/CD	TeamCity, Jenkins, GitLab, GoCD, Azure DevOps, GitHub Actions, Bitbucket CI …
Удаленный репозиторий	Bitbucket, Git, GitLab …
Работа с git	JGit, Pygit2, GitPython, GO-Git …
Фреймворки для автотестов	JUnit, Pytest, TestNG …

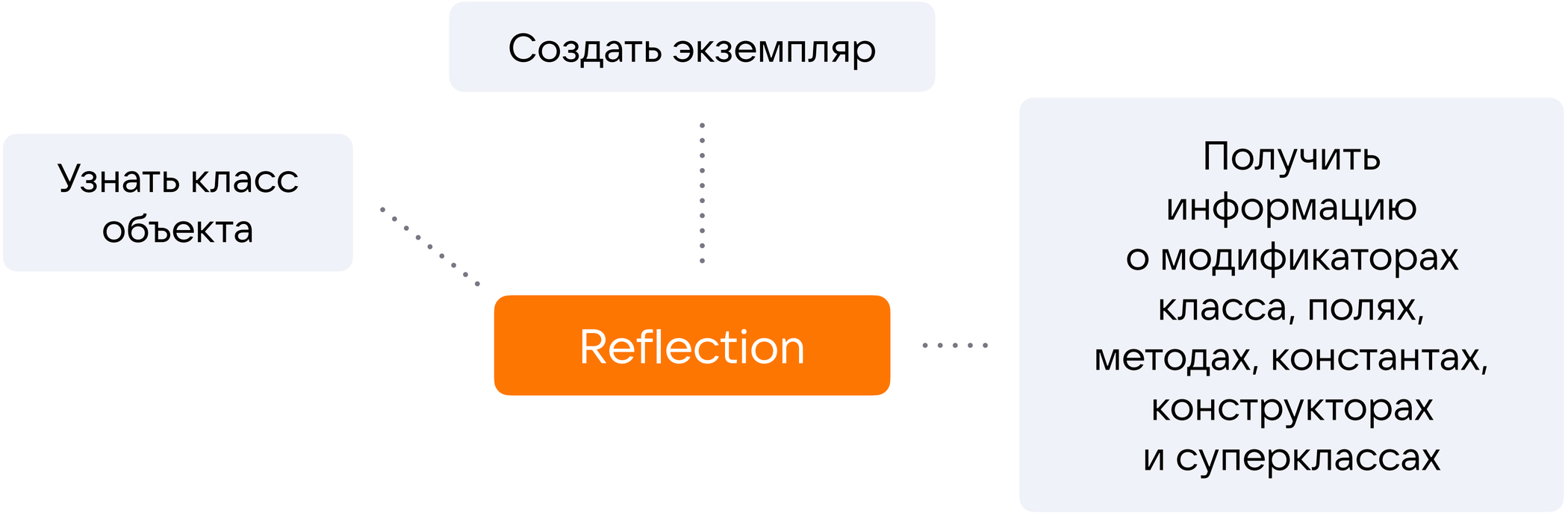
Языки программирования с Reflection	Java, Python, Kotlin, Go …
API клиент	Любой API клиент написанный на языках, у которых есть механизм Reflection
CI/CD	TeamCity, Jenkins, GitLab, GoCD, Azure DevOps, GitHub Actions, Bitbucket CI …
Удаленный репозиторий	Bitbucket, Git, GitLab …
Работа с git	JGit, Pygit2, GitPython, GO-Git …
Фреймворки для автотестов	JUnit, Pytest, TestNG …

Reflection

Узнать класс объекта

Reflection


Создать экземпляр
Узнать класс объекта
Reflection


Создать экземпляр
Узнать класс объекта
Reflection
Получить информацию о модификаторах класса, полях, методах, константах, конструкторах и суперклассах


Создать экземпляр
Узнать класс объекта
Reflection
Получить информацию о модификаторах класса, полях, методах, константах, конструкторах и суперклассах
Вызвать метод объекта по имени


Создать экземпляр
Узнать класс объекта
Получить информацию о модификаторах класса, полях, методах, константах, конструкторах и суперклассах
Reflection
Получить и установить значение поля объекта по имени
Вызвать метод объекта по имени

Языки программирования с Reflection	Java, Python, Kotlin, Go …
API клиент	Любой API клиент написанный на языках, у которых есть механизм Reflection
CI/CD	TeamCity, Jenkins, GitLab, GoCD, Azure DevOps, GitHub Actions, Bitbucket CI …
Удаленный репозиторий	Bitbucket, Git, GitLab …
Работа с git	JGit, Pygit2, GitPython, GO-Git …
Фреймворки для автотестов	JUnit, Pytest, TestNG …

Языки программирования с Reflection	Java, Python, Kotlin, Go …
API клиент	Любой API клиент написанный на языках, у которых есть механизм Reflection
CI/CD	TeamCity, Jenkins, GitLab, GoCD, Azure DevOps, GitHub Actions, Bitbucket CI …
Удаленный репозиторий	Bitbucket, Git, GitLab …
Работа с git	JGit, Pygit2, GitPython, GO-Git …
Фреймворки для автотестов	JUnit, Pytest, TestNG …

Языки программирования с Reflection	Java, Python, Kotlin, Go …
API клиент	Любой API клиент написанный на языках, у которых есть механизм Reflection
CI/CD	TeamCity, Jenkins, GitLab, GoCD, Azure DevOps, GitHub Actions, Bitbucket CI …
Удаленный репозиторий	Bitbucket, Git, GitLab …
Работа с git	JGit, Pygit2, GitPython, GO-Git …
Фреймворки для автотестов	JUnit, Pytest, TestNG …

Языки программирования с Reflection	Java, Python, Kotlin, Go …
API клиент	Любой API клиент написанный на языках, у которых есть механизм Reflection
CI/CD	TeamCity, Jenkins, GitLab, GoCD, Azure DevOps, GitHub Actions, Bitbucket CI …
Удаленный репозиторий	Bitbucket, Git, GitLab …
Работа с git	JGit, Pygit2, GitPython, GO-Git …
Фреймворки для автотестов	JUnit, Pytest, TestNG …

Языки программирования с Reflection	Java, Python, Kotlin, Go …
API клиент	Любой API клиент написанный на языках, у которых есть механизм Reflection
CI/CD	TeamCity, Jenkins, GitLab, GoCD, Azure DevOps, GitHub Actions, Bitbucket CI …
Удаленный репозиторий	Bitbucket, Git, GitLab …
Работа с git	JGit, Pygit2, GitPython, GO-Git …
Фреймворки для автотестов	JUnit, Pytest, TestNG …

Наш стек

Языки программирования с Reflection	Java
API клиент	Кастомный API клиент
CI/CD	TeamCity
Удаленный репозиторий	Bitbucket
Работа с git	JGit
Фреймворки для автотестов	JUnit5

Список методов

Комментарий к видео	**Загрузка видео**
Параметры: 1) Видео ID — обязательный параметр 2) Текст — обязательный параметр	**Параметры:** 1) Название - необязательный параметр 2) Путь к файлу - обязательный параметр 3) Дополнительная информация — необязательный параметр
Тип сессии: Анонимная сессия	**Тип сессии:** Пользовательская сессия
Просмотр видео	**Отправка другу видео**
Параметры: 1) Видео ID — обязательный параметр	**Параметры:** 1) ID друга — обязательный параметр 2) ID видео — обязательный параметр 3) Текст — необязательный параметр
Тип сессии: Без сессии	**Тип сессии:** Пользовательская сессия

OK.TECH

ВЖЖЖУХ

```
@ApiMethodEndpoint(
ApiMethod.COMMENT_HEISENBUG_VIDEO
)
commentHeisenbugVideo(
@RestParam(value = "videoId", required = true)
Long var1,
@RestParam(value = "text", required = true)
String var2
)
```

```
@ApiMethodEndpoint(
ApiMethod.UPLOAD_HEISENBUG_VIDEO
)
uploadHeisenbugVideo(
@RestParam(value = "name", required = false)
String var1,
@RestParam(value = "pathToFile", required =
true)
File var2,
@RestParam(value = "info", required = false)
VideoInfo var3
)
```

```
@ApiMethodEndpoint(
ApiMethod.WATCH_HEISENBUG_VIDEO
)
watchHeisenbugVideo(
@RestParam(value = "videoId", required = true)
Long var1
)
```

```
@ApiMethodEndpoint(
ApiMethod.SEND_HEISENBUG_VIDEO
)
sendFriendHeisenbugVideo(
@RestParam(value = "friendId", required = true)
Long var1,
@RestParam(value = "videoId", required = true)
Long var2,
@RestParam(value = "text", required = false)
String var3
)
```

```
@ApiMethodEndpoint(
ApiMethod.COMMENT_HEISENBUG_VIDEO
)
commentHeisenbugVideo(
@RestParam(value = "videoId", required = true)
Long var1,
@RestParam(value = "text", required = true)
String var2
)
```

```
@ApiMethodEndpoint(
ApiMethod.UPLOAD_HEISENBUG_VIDEO
)
uploadHeisenbugVideo(
@RestParam(value = "name", required = false)
String var1,
@RestParam(value = "pathToFile", required =
true)
File var2,
@RestParam(value = "info", required = false)
VideoInfo var3
)
```

```
@ApiMethodEndpoint(
ApiMethod.WATCH_HEISENBUG_VIDEO
)
watchHeisenbugVideo(
@RestParam(value = "videoId", required = true)
Long var1
)
```

```
@ApiMethodEndpoint(
ApiMethod.SEND_HEISENBUG_VIDEO
)
sendFriendHeisenbugVideo(
@RestParam(value = "friendId", required = true)
Long var1,
@RestParam(value = "videoId", required = true)
Long var2,
@RestParam(value = "text", required = false)
String var3
)
```

```
@ApiMethodEndpoint(
ApiMethod.COMMENT_HEISENBUG_VIDEO(
"heisenbug",
"commentHeisenbugVideo",
ApiProtectionLevel.ANONYM_SESSION_REQUIRED
)
)
```

```
@ApiMethodEndpoint(
ApiMethod.UPLOAD_HEISENBUG_VIDEO(
"heisenbug",
"uploadHeisenbugVideo",
ApiProtectionLevel.SESSION_REQUIRED
)
)
```

```
@ApiMethodEndpoint(
ApiMethod.WATCH_HEISENBUG_VIDEO(
"heisenbug",
"watchHeisenbugVideo",
ApiProtectionLevel.SESSION_PROHIBITED
)
)
```

```
@ApiMethodEndpoint(
ApiMethod.SEND_HEISENBUG_VIDEO(
"heisenbug",
"sendHeisenbugVideo",
ApiProtectionLevel.SESSION_REQUIRED
)
)
```

```
method.getAnnotation(ApiMethodEndpoint.class).value().getCategory();
```

`@ApiMethodEndpoint(`
`ApiMethod.COMMENT_HEISENBUG_VIDEO(`
`"heisenbug",`
`"commentHeisenbugVideo",`
`ApiProtectionLevel.ANONYM_SESSION_REQUIRED`
`)`
`)`
	`@ApiMethodEndpoint(`
	`ApiMethod.UPLOAD_HEISENBUG_VIDEO(`
	`"heisenbug",`
	`"uploadHeisenbugVideo",`
	`ApiProtectionLevel.SESSION_REQUIRED`
	`)`
	`)`

`@ApiMethodEndpoint(`
`ApiMethod.WATCH_HEISENBUG_VIDEO(`
`"heisenbug",`
`"watchHeisenbugVideo",`
`ApiProtectionLevel.SESSION_PROHIBITED`
`)`
`)`
	`@ApiMethodEndpoint(`
	`ApiMethod.SEND_HEISENBUG_VIDEO(`
	`"heisenbug",`
	`"sendHeisenbugVideo",`
	`ApiProtectionLevel.SESSION_REQUIRED`
	`)`
	`)`

```
method.getAnnotation(ApiMethodEndpoint.class).value().getMethodName();
```

```
@ApiMethodEndpoint(
ApiMethod.COMMENT_HEISENBUG_VIDEO(
"heisenbug",
"commentHeisenbugVideo",
ApiProtectionLevel.ANONYM_SESSION_REQUIRED
)
)
```

```
@ApiMethodEndpoint(
ApiMethod.UPLOAD_HEISENBUG_VIDEO(
"heisenbug",
"uploadHeisenbugVideo",
ApiProtectionLevel.SESSION_REQUIRED
)
)
```

```
@ApiMethodEndpoint(
ApiMethod.WATCH_HEISENBUG_VIDEO(
"heisenbug",
"watchHeisenbugVideo",
ApiProtectionLevel.SESSION_PROHIBITED
)
)
```

```
@ApiMethodEndpoint(
ApiMethod.SEND_HEISENBUG_VIDEO(
"heisenbug",
"sendHeisenbugVideo",
ApiProtectionLevel.SESSION_REQUIRED
)
)
```

```
method.getAnnotation(ApiMethodEndpoint.class).value().getProtectionLevel();
```

```
@ApiMethodEndpoint(
ApiMethod.COMMENT_HEISENBUG_VIDEO(
"heisenbug",
"commentHeisenbugVideo",
ApiProtectionLevel.ANONYM_SESSION_REQUIRED
)
)
```

```
@ApiMethodEndpoint(
ApiMethod.UPLOAD_HEISENBUG_VIDEO(
"heisenbug",
"uploadHeisenbugVideo",
ApiProtectionLevel.SESSION_REQUIRED
)
)
```

```
@ApiMethodEndpoint(
ApiMethod.WATCH_HEISENBUG_VIDEO(
"heisenbug",
"watchHeisenbugVideo",
ApiProtectionLevel.SESSION_PROHIBITED
)
)
```

```
@ApiMethodEndpoint(
ApiMethod.SEND_HEISENBUG_VIDEO(
"heisenbug",
"sendHeisenbugVideo",
ApiProtectionLevel.SESSION_REQUIRED
)
)
```

```
@Tag("heisenbug")
@Tag("commentHeisenbugVideo")
bindAnonymSession();

testHeisenbugCommentHeisenbugVideo + тип
проверяемой сессии
```

```
@Tag("heisenbug")
@Tag("uploadHeisenbugVideo")
bindNewSession(testBot);

testHeisenbugUploadHeisenbugVideo + тип
проверяемой сессии
```

```
@Tag("heisenbug")
@Tag("watchHeisenbugVideo")

testHeisenbugWatchHeisenbugVideo + тип
проверяемой сессии
```

```
@Tag("heisenbug")
@Tag("sendHeisenbugVideo")
bindNewSession(testBot);

testHeisenbugSendHeisenbugVideo + тип
проверяемой сессии
```

commentHeisenbugVideo(
`@RestParam(value = "videoId", required = true)`
Long var1,
`@RestParam(value = "text", required = true)`
String var2
)

uploadHeisenbugVideo(
`@RestParam(value = "name", required = false)`
String var1,
`@RestParam(value = "pathToFile", required = true)`
File var2,
`@RestParam(value = "info", required = false)`
VideoInfo var3
)

watchHeisenbugVideo(
`@RestParam(value = "videoId", required = true)`
Long var1
)

sendFriendHeisenbugVideo(
`@RestParam(value = "friendId", required = true)`
Long var1,
`@RestParam(value = "videoId", required = true)`
Long var2,
`@RestParam(value = "text", required = false)`
String var3
)

```
"apiClient."
+ getServiceMethodByNameService(method.getDeclaringClass().getSimpleName()
+ "()."
+ method.getName();
```

commentHeisenbugVideo(
`@RestParam(value = "videoId", required = true)`
Long var1,
`@RestParam(value = "text", required = true)`
String var2
)
	uploadHeisenbugVideo(
	`@RestParam(value = "name", required = false)`
	String var1,
	`@RestParam(value = "pathToFile", required = true)`
	File var2,
	`@RestParam(value = "info", required = false)`
	VideoInfo var3
	)

watchHeisenbugVideo(
`@RestParam(value = "videoId", required = true)`
Long var1
)
	sendFriendHeisenbugVideo(
	`@RestParam(value = "friendId", required = true)`
	Long var1,
	`@RestParam(value = "videoId", required = true)`
	Long var2,
	`@RestParam(value = "text", required = false)`
	String var3
	)

`apiClient.getHeisenbugService()` `  .commentHeisenbugVideo(…);`	`apiClient.getHeisenbugService()` `  .uploadHeisenbugVideo(…);`
`apiClient.getHeisenbugService()` `  .watchHeisenbugVideo(…);`	`apiClient.getHeisenbugService()` `  .sendHeisenbugVideo(…);`

commentHeisenbugVideo(
```
@RestParam(value = "videoId", required = true)
Long var1,
@RestParam(value = "text", required = true)
String var2
```
)

uploadHeisenbugVideo(
```
@RestParam(value = "name", required = false)
String var1,
@RestParam(value = "pathToFile", required =
true)
File var2,
@RestParam(value = "info", required = false)
VideoInfo var3
```
)

watchHeisenbugVideo(
```
@RestParam(value = "videoId", required = true)
Long var1
```
)

sendFriendHeisenbugVideo(
```
@RestParam(value = "friendId", required = true)
Long var1,
@RestParam(value = "videoId", required = true)
Long var2,
@RestParam(value = "text", required = false)
String var3
```
)

```
parameter.getAnnotation(RestParam.class).value();
```

`@RestParam(value = "videoId", required = true)`
Long var1,
`@RestParam(value = "text", required = true)`
String var2
	`@RestParam(value = "name", required = false)`
	String var1,
	`@RestParam(value = "pathToFile", required = true)`
	File var2,
	`@RestParam(value = "info", required = false)`
	VideoInfo var3

`@RestParam(value = "videoId", required = true)`
Long var1
	`@RestParam(value = "friendId", required = true)`
	Long var1,
	`@RestParam(value = "videoId", required = true)`
	Long var2,
	`@RestParam(value = "text", required = false)`
	String var3

```
parameter.getAnnotation(RestParam.class).required();
```

`@RestParam(value = "videoId", required = true)`
Long var1,
`@RestParam(value = "text", required = true)`
String var2
	`@RestParam(value = "name", required = false)`
	String var1,
	`@RestParam(value = "pathToFile", required = true)`
	File var2,
	`@RestParam(value = "info", required = false)`
	VideoInfo var3

`@RestParam(value = "videoId", required = true)`
Long var1
	`@RestParam(value = "friendId", required = true)`
	Long var1,
	`@RestParam(value = "videoId", required = true)`
	Long var2,
	`@RestParam(value = "text", required = false)`
	String var3

```
parameter.getType().getSimpleName();
```

`@RestParam(value = "videoId", required = true)`
Long var1,
`@RestParam(value = "text", required = true)`
String var2
	`@RestParam(value = "name", required = false)`
	String var1,
	`@RestParam(value = "pathToFile", required = true)`
	File var2,
	`@RestParam(value = "info", required = false)`
	VideoInfo var3

`@RestParam(value = "videoId", required = true)`
Long var1
	`@RestParam(value = "friendId", required = true)`
	Long var1,
	`@RestParam(value = "videoId", required = true)`
	Long var2,
	`@RestParam(value = "text", required = false)`
	String var3

```
@RestParam(value = "videoId", required = true)
Long var1,
@RestParam(value = "text", required = true)
String var2

apiClient.getHeisenbugService().commentHeisenbugVideo((Long) 0, (String) "1234567");
```

```
@RestParam(value = "name", required = false)
String var1,
@RestParam(value = "pathToFile", required = true)
File var2,
@RestParam(value = "info", required = false)
VideoInfo var3

apiClient.getHeisenbugService().uploadHeisenbugVideo((String) null,(File) new File(), (VidoInfo) null);
```

```
@RestParam(value = "videoId", required = true)
Long var1

apiClient.getHeisenbugService().watchHeisenbugVideo((Long) 0);
```

```
@RestParam(value = "friendId", required = true)
Long var1,
@RestParam(value = "videoId", required = true)
Long var2,
@RestParam(value = "text", required = false)
String var3

apiClient.getHeisenbugService().sendHeisenbugVideo((Long) 0,(Long) 0,(String) null);
```

Дефолтные значения

Список для примитивных типов данных	Список для ссылочных типов данных

`switch (parameter.getType().getSimpleName()) {`
`  case "byte" -> "(byte) 0";`
`  case "short" -> "(short) 0";`
`  case "int", "long" -> "0";`
`  case "float", "double" -> "0.0";`
`  case "boolean" -> "false";`
`  default -> "null";`
`}`
	`String type = "";`
	`switch (parameter.getType().getSimpleName()) {`
	`  case "Double" -> type = "0.0";`
	`  case "Float" -> type = "0.0F";`
	`  case "Short" -> type = "(short) 0";`
	`  case "Byte" -> type = "(byte) 0";`
	`  case "Integer" -> type = "0";`
	`  case "Long" -> type = "0L";`
	`  case "Boolean" -> type = "false";`
	`  case "String" -> type = "\"1234567\"";`
	`  case "File" -> type = "new File()»;`
	`  case "VideoInfo" -> type =`
	`  «new VideoInfo(«12345», true)»;`
	`  ...`
	`  ...`
	`  ...`
	`  default -> type = "null";`
	`}`
	`return type;`
	`}`

Какие тесты мы можем генерировать?

- Вызовы методов в некорректных сессиях
- Вызовы метода без одного обязательного параметра
- Вызовы метода без нескольких обязательных параметров

Базовый шаблон теста

```
@аннотации
@теги
название метода (параметры метода) {
    логи;

    тип сессии;

    проверка;

    логи;
}
```

Вызов без сессии

```
@Test
@Tag("commentHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugCommentHeisenbugVideoNoSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.commentHeisenbugVideo' без сессии»);
    noSession();

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().commentHeisenbugVideo(0, "1234567"),
        throwsException(ApiException.class, containsString(ERROR_NO_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.commentHeisenbugVideo' без сессии");
}
```

Вызов без сессии

```
@Test
@Tag("commentHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugCommentHeisenbugVideoNoSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.commentHeisenbugVideo' без сессии");
    noSession();

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().commentHeisenbugVideo(0, "1234567"),
        throwsException(ApiException.class, containsString(ERROR_NO_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.commentHeisenbugVideo' без сессии");
}
```

Вызов без сессии

```
@Test
@Tag("commentHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugCommentHeisenbugVideoNoSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.commentHeisenbugVideo' без сессии");
    noSession();

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().commentHeisenbugVideo(0, "1234567"),
        throwsException(ApiException.class, containsString(ERROR_NO_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.commentHeisenbugVideo' без сессии");
}
```

Вызов без сессии

```
@Test
@Tag("commentHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugCommentHeisenbugVideoNoSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.commentHeisenbugVideo' без сессии»);
    noSession();

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().commentHeisenbugVideo(0, "1234567"),
        throwsException(ApiException.class, containsString(ERROR_NO_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.commentHeisenbugVideo' без сессии");
}
```

Вызов без сессии

```
@Test
@Tag("commentHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugCommentHeisenbugVideoNoSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.commentHeisenbugVideo' без сессии»);
    noSession();

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().commentHeisenbugVideo(0, "1234567"),
        throwsException(ApiException.class, containsString(ERROR_NO_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.commentHeisenbugVideo' без сессии");
}
```

Вызов без сессии

```
@Test
@Tag("commentHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugCommentHeisenbugVideoNoSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.commentHeisenbugVideo' без сессии»);
    noSession();

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().commentHeisenbugVideo(0, "1234567"),
        throwsException(ApiException.class, containsString(ERROR_NO_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.commentHeisenbugVideo' без сессии");
}
```

Вызов без сессии

```
@Test
@Tag("commentHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugCommentHeisenbugVideoNoSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.commentHeisenbugVideo' без сессии»);
    noSession();

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().commentHeisenbugVideo(0, "1234567"),
        throwsException(ApiException.class, containsString(ERROR_NO_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.commentHeisenbugVideo' без сессии");
}
```

Вызов из анонимной сессии

```
@Test
@Tag("uploadHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugUploadHeisenbugVideoAnonymSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.uploadHeisenbugVideo' в анонимной сессии");
    bindAnonymSession();

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().uploadHeisenbugVideo(null, new File(), null),
        throwsException(ApiException.class, containsString(ERROR_ANONYM_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.uploadHeisenbugVideo' в анонимной сессии");
}
```

Вызов из анонимной сессии

```
@Test
@Tag("uploadHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugUploadHeisenbugVideoAnonymSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.uploadHeisenbugVideo' в анонимной сессии");
    bindAnonymSession();

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().uploadHeisenbugVideo(null, new File(), null),
        throwsException(ApiException.class, containsString(ERROR_ANONYM_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.uploadHeisenbugVideo' в анонимной сессии");
}
```

Вызов в пользовательской сессии

```
@Test
@Tag("watchHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugWatchHeisenbugVideoUserSession(TestBot testBot) {
    LOGGER.intention("Проверим работу метода heisenbug.watchHeisenbugVideo в пользовательской сессии");
    bindNewSession(testBot);

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().watchHeisenbugVideo(0),
        throwsException(ApiException.class, containsString(ERROR_USER_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода heisenbug.watchHeisenbugVideo в пользовательской сессии");
}
```

Вызов в пользовательской сессии

```
@Test
@Tag("watchHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugWatchHeisenbugVideoUserSession(TestBot testBot) {
    LOGGER.intention("Проверим работу метода heisenbug.watchHeisenbugVideo в пользовательской сессии");
    bindNewSession(testBot);

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().watchHeisenbugVideo(0),
        throwsException(ApiException.class, containsString(ERROR_USER_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода heisenbug.watchHeisenbugVideo в пользовательской сессии");
}
```

Вызов без одного обязательного параметра

```
@Test
@Tag("uploadHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugUploadHeisenbugVideoOneRequiredParameter(TestBot testBot) {
    LOGGER.intention("Проверим работу метода 'heisenbug.uploadHeisenbugVideo' без единственного обязательного параметра");
    bindNewSession(testBot);

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().uploadHeisenbugVideo(null, null, null),
        throwsException(ApiException.class, containsString(ONE_PARAMETER_ERROR))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.uploadHeisenbugVideo' без единственного обязательного параметра");
}
```

Вызов без одного обязательного параметра

```
@Test
@Tag("uploadHeisenbugVideo")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugUploadHeisenbugVideoOneRequiredParameter(TestBot testBot) {
    LOGGER.intention("Проверим работу метода 'heisenbug.uploadHeisenbugVideo' без единственного обязательного параметра");
    bindNewSession(testBot);

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().uploadHeisenbugVideo(null, null, null),
        throwsException(ApiException.class, containsString(ONE_PARAMETER_ERROR))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.uploadHeisenbugVideo' без единственного обязательного параметра");
}
```

Вызов без нескольких обязательных параметров

```
@Tag("sendFriendHeisenbug")
@Tag("badpass") @Tag("generated_tests")
@MethodSource("provideMissingParams")
@ParameterizedTest(name = "[{index}] friendId: {0}, videoId: {1}, text: {2}, errorMessage: {3}")
public void testSendFriendHeisenbugWithMissingParams(TestBot testBot, Long friendId, Long videoId, String errorMessage) {
    LOGGER.intention("Проверим работу метода 'heisenbug.sendFriendHeisenbug' при отсутствии обязательных параметров");
    bindNewSession(testBot);

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().sendFrinedHeisenbug(friendId, videoId, null),
        throwsException(ApiException.class, containsString(errorMessage))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.sendFriendHeisenbug' при отсутствии обязательных параметров");
}
```

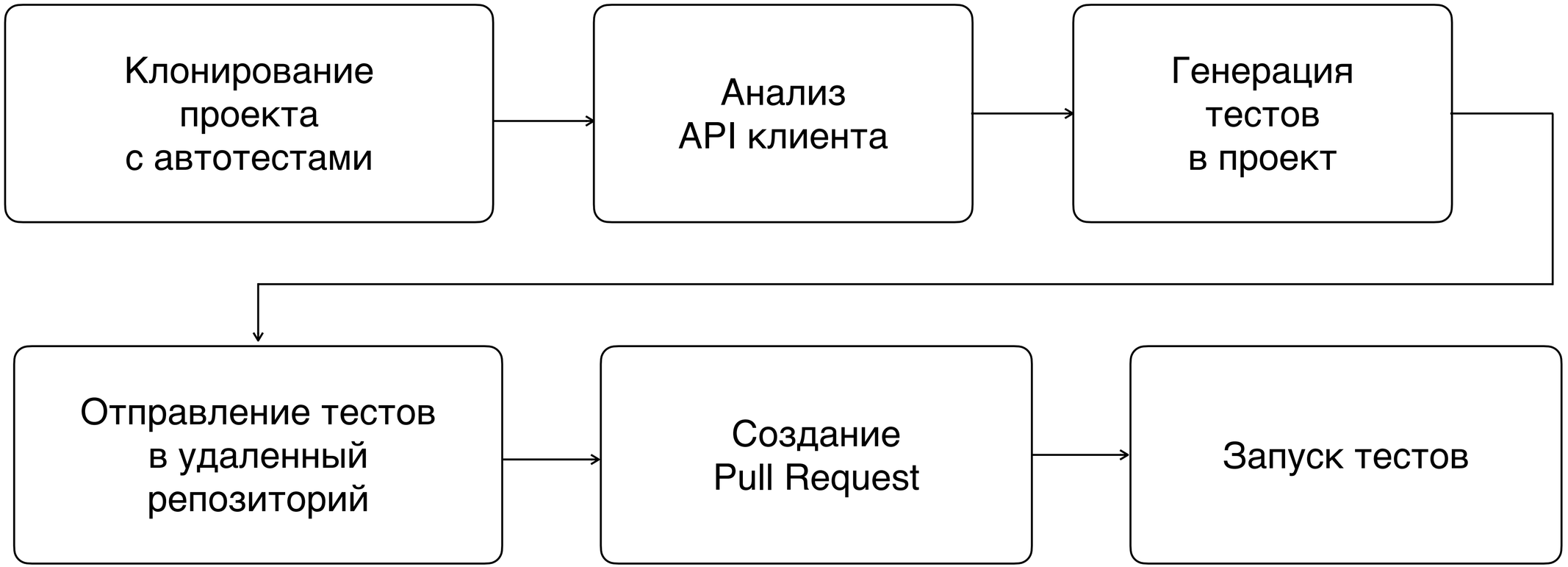
Вызов без нескольких обязательных параметров

```
@Tag("sendFriendHeisenbug")
@Tag("badpass") @Tag("generated_tests")
@MethodSource("provideMissingParams")
@ParameterizedTest(name = "[{index}] friendId: {0}, videoId: {1}, text: {2}, errorMessage: {3}")
public void testSendFriendHeisenbugWithMissingParams(TestBot testBot, Long friendId, Long videoId, String errorMessage) {
    LOGGER.intention("Проверим работу метода 'heisenbug.sendFriendHeisenbug' при отсутствии обязательных параметров");
    bindNewSession(testBot);

    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().sendFrinedHeisenbug(friendId, videoId, null),
        throwsException(ApiException.class, containsString(errorMessage))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.sendFriendHeisenbug' при отсутствии обязательных параметров");
}
```

Вызов без нескольких обязательных параметров

```
@Tag("sendFriendHeisenbug")
@Tag("badpass")
@Tag("generated_tests")
@MethodSource("provideMissingParams")
@ParameterizedTest(name = "[{index}] friendId: {0}, videoId: {1}, text: {2}, errorMessage: {3}")
public void testSendFriendHeisenbugWithMissingParams(TestBot testBot, Long friendId, Long
videoId, String errorMessage) { ... }

private static Stream<Arguments> provideMissingParams() {
    return Stream.of(
            Arguments.of(null, null, FRIEND_ID_MISSING_PARAM),
            Arguments.of(0, null, VIDEO_ID_MISSING_PARAM)
    );
}
```

Шаги генерации тестов



Клонирование репозитория

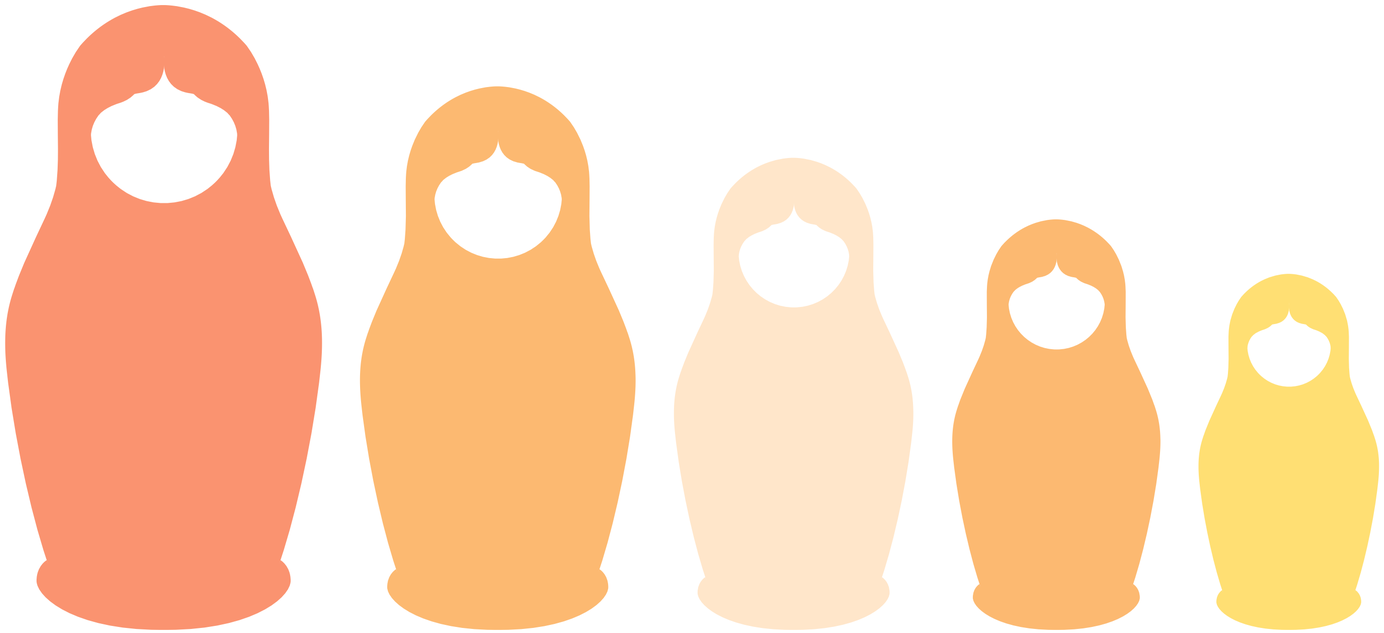
Консоль	Код
git clone ssh://name.git	Git git = Git.cloneRepository() .setURI("ssh://name.git") .setDirectory(new File("tmp/autotest-api")) .call();
git checkout имя-ветки	git.checkout() .setName(branch) .call();

Анализ API клиента

Собираем список API методов:

```
ApiClient.class.getMethods();
```


Генерация тестов



1 2 **3** 4 5 6

Генерируем assert

```
assertThat(

);
```

1 2 3 4 5 6

Генерируем assert

```
assertThat(
    "Не получили ожидаемую ошибку при вызове метода",


);
```


Генерируем assert

```
assertThat(
    "Не получили ожидаемую ошибку при вызове метода",
    () -> apiClient.getHeisenbugService().uploadHeisenbugVideo(null, null, null),

);
```

1 2 3 4 5 6

Генерируем assert

```
assertThat(
    "Не получили ожидаемую ошибку при вызове метода",
    () -> apiClient.getHeisenbugService().uploadHeisenbugVideo(null, null, null),
    throwsException(ApiException.class, containsString(ERROR_NO_SESSION))
);
```


Генерируем метод

```
public void testHeisenbugUploadHeisenbugVideoNoSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.uploadHeisenbugVideo' без сессии");


    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().uploadHeisenbugVideo(null, null, null),
        throwsException(ApiException.class, containsString(ERROR_NO_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода heisenbug.uploadHeisenbugVideo без
сессии");
}
```

1 2 3 4 5 6

Генерируем теги

```
@Test
@Tag("heisenbug")
@Tag("badpass")
@Tag("generated_tests")
public void testHeisenbugUploadHeisenbugVideoNoSession() {
    LOGGER.intention("Проверим работу метода 'heisenbug.uploadHeisenbugVideo' без сессии");


    assertThat(
        "Не получили ожидаемую ошибку при вызове метода",
        () -> apiClient.getHeisenbugService().uploadHBVideo(null, null, null),
        throwsException(ApiException.class, containsString(ERROR_NO_SESSION))
    );

    LOGGER.success("Успешно проверили работу метода 'heisenbug.uploadHeisenbugVideo' без
сессии");
}
```


Генерируем тестовый класс

```
package ...

import  ...

/**
* Автосгенерированный тест на метод heisenbug.uploadHeisenbugVideo
* Документация: ...
*/
public class TestHeisenbugUploadHeisenbugVideoBadPass extends TestBase {
    private static final String ERROR_NO_SESSION = "SESSION_REQUIRED";

    @Test
    @Tag("heisenbug")
    @Tag("badpass")
    @Tag("generated_tests")
    public void testHeisenbugUploadHeisenbugVideoNoSession() { ... }
}
```

1 2 3 4 5 6

Отправка тестов в удаленный репозиторий

Консоль	Код
git clone ssh://name.git	Git git = Git.*cloneRepository*() .setURI("ssh://name.git") .setDirectory(new File("tmp/autotest-api")) .call();
git checkout имя-ветки	git.checkout() .setName(branch) .call();
git add "." **git commit -m message** **git push**	**git.add().addFilepattern(".").call();** **git.commit().setMessage(message).call();** **git.push().call();**

Создание Pull Request

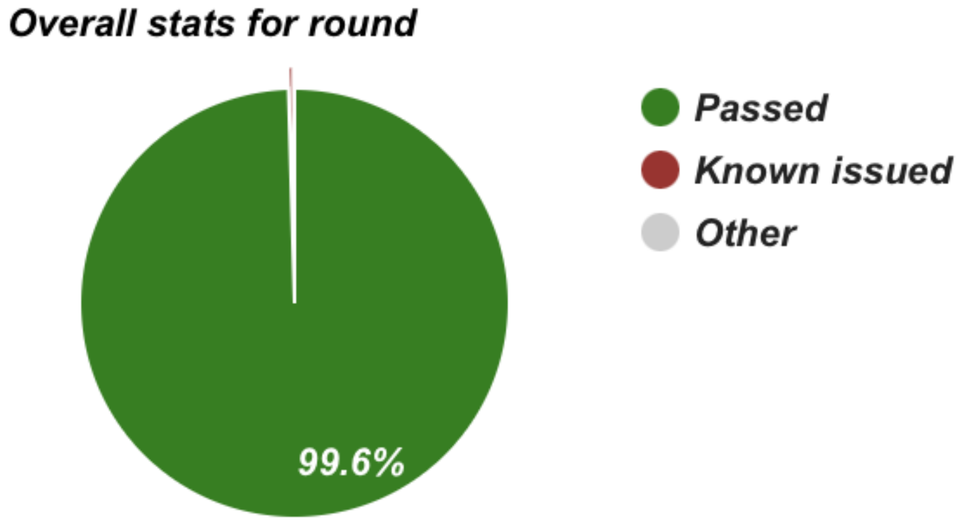
```
CreatePullRequest createPullRequest = new CreatePullRequest(
    title,                                                          Передаём название
    DESCRIPTION + forticomVersion + DESC_PART_2,                    Передаём описание
    new BranchRef(MAIN_BRANCH, BranchType.BRANCH),                  Указываем ветку, в которую будем заливать изменения
    new BranchRef(fromBranchName, BranchType.BRANCH)                Указываем ветку, из которой хотим заливать изменения
);

stashApi.createPullRequest(PROJECT_KEY, REPOSITORY_SLUG, createPullRequest);
```

https://developer.atlassian.com/cloud/bitbucket/rest/api-group-pullrequests/#api-repositories-workspace-repo-slug-pullrequests-post

1 2 3 4 **5** 6

OK.TECH

Собственный сервис для запуска автотестов



Какую пользу мы получили

→ Увеличение покрытия автотестами на 25%

Какую пользу мы получили

- → Увеличение покрытия автотестами на 25%
- → Сокращение техдолга

Какую пользу мы получили

- → Увеличение покрытия автотестами на 25%
- → Сокращение техдолга
- → Больше времени на сложные ручные кейсы

Какую пользу мы получили

- Увеличение покрытия автотестами на 25%
- Сокращение техдолга
- Больше времени на сложные ручные кейсы
- Актуальность автотестов

Какую пользу мы получили

- Увеличение покрытия автотестами на 25%
- Сокращение техдолга
- Больше времени на сложные ручные кейсы
- Актуальность автотестов
- Быстрое изменение кода автосгенерированных тестов

Какие тесты мы будем генерировать?

→ Проверка пермиссий приложений

Какие тесты мы будем генерировать?

- Проверка пермиссий приложений
- Проверка, что нет ошибок при указании обязательных/необязательных параметров

Какие тесты мы будем генерировать?

- Проверка пермиссий приложений
- Проверка, что нет ошибок при указании обязательных/необязательных параметров
- Проверка граничных значений параметров

Какие тесты мы будем генерировать?

- Проверка пермиссий приложений
- Проверка, что нет ошибок при указании обязательных/необязательных параметров
- Проверка граничных значений параметров
- Проверка базовых кейсов бизнес логики

OK.TECH

Спасибо за внимание)))

Елизавета @riiinj
Николай @doom800300