Как выглядит Roadmap тестировщика

 Тестировщик — многофункциональная фигура. Специалист, получив базовые знания, постепенно развивает свои навыки в разных направлениях. Он может выбрать одно или несколько таких направлений в зависимости от специфики компании и других факторов.

     Рассмотрим скиллы и технологии, которыми должен/может владеть тестировщик от элементарных до самых продвинутых.

**1.Quality Assurance Fundamentals — обеспечение качества ПО:**

      Теоретическая база тестировщика, в которую входят знания о багах, тест-кейсах, test-suites, планах тестирования, программах управления тест-кейсами, функциональном и нефункциональном тестировании и т.д.;

**2. SDLC или жизненный цикл ПО:**

     Глубокое понимания самого цикла разработки продукта, в котором участвуют десятки специалистов. Базовые или продвинутые знания о модели “Водопада”, V-модели, спиральной модели, основах методологий Scrum, Kanban;

**3. Интернет-технологии:**

     Чтобы понять, как работает ПО, важно ознакомиться с ключевыми технологиями Интернета. Необходимо иметь представления об IP, DNS, HTTP, OSI, HTML, CSS, JS и т.д.

**4. Computer Science Fundamentals:**

     Для тестировщика релевантны не все технологии компьютерной инженерии. В данном разделе важно понимать:

* Как “железо” компьютера управляется софтом;
* Как создается ПО;
* Особенности разноуровневых языков программирования;
* Алгоритмы, основы анализа данных, многопоточность современного программирования.

     Для тестировщика изучение языков программирования является задачей второго плана. В первую очередь стоит делать упор на автоматизацию тестирования. Именно владение концепциями разработки ПО дает возможность в любой момент быстро овладеть нужным языком;

**5. Веб-приложения:**

     Отсюда начинается “второй” уровень развития тестировщика, когда все базовые навыки развиты на хорошем уровне. Хотя веб-интерфейс характерен не для всех цифровых продуктов, тестировщику необходимы приемлемые знания веб-разработки:

* HTML, CSS, JS — база;
* AJAX, SWA и PWA-приложения;
* Базовая криптография, системы управления контентом;
* Адаптивные/реактивные приложения;
* Фреймворки React, Angular;
* Тестирование “под капотом” юнит-тестами, в headless-браузере.

**6. Programming:**

     Программированием должен владеть любой современный тестировщик среднего и высокого уровня, без этих скиллов дальнейшее карьерное развитие просто невозможно. Данная тема крайне обширна, и выбор языка зависит от направления, в котором Вы планируете работать. И все же стоит изучить как минимум на базовом уровне несколько языков:

* JavaScript;
* TypeScript;
* Java;
* C#;
* Python.

**7. Enterprise Architecture:**

      Достаточно сложная и обширная область, которая включает такие темы:

* REST-сервисы;
* Микросервисная архитектура;
* Трехслойная архитектура;
* Реляционные и объектные хранилища;
* Управление кэшем, прокси, репликация и многое другое.

       Каждая из них в целом независима друг от друга, может изучаться в различном порядке. Для понимания их функционирования важно знать, что такое IaaS, PaaS, SaaS, AWS, GCP, Azure.

**8. Автоматизация тестирования:**

       Эта сфера деятельности невозможна в первую очередь без хороших знаний программирования и энтерпрайз-архитектур. Написание ПО для автоматизации процессов тестирования относится к продвинутым скиллам тестировщика.

     Что здесь важно знать и понимать?

* Для чего пишутся такие виды тестов;
* Концепции “пирамида”, oracles, test surfaces;
* low-code, no-code, BDD-языки;
* Инструменты WireMock, Montebank.

     Автоматизация — большая категория деятельности, которая также подразделяется на:

* Unit-тестирование;
* Тестирование API;
* Тестирование интерфейсов.

**9. Эджайл (AGILE):**

     Гибкая методология разработки AGILE — индивидуальная для каждой команды разработки. Для тестировщика важно именно понимание концепции деятельности подразделения.

     Важные термины в Roadmap тестировщика:

* Story definition;
* Story estimation;
* Acceptance criteria definition;
* Story estimation techniques;
* Церемонии: stand up, demo, retro.

     Стандартные средства управления:

* Jira;
* Rally;
* MS Project.

**10. Другие виды тестирования:**

     Знания в каждой из областей тестирования не помешают. Но, как правило, в каждой команде есть отдельный специалист под конкретную задачу:

* Тестирование производительности — знание типов и номенклатуры тестов, рабочих инструментов, интеграцию в пайплайны CI|CD. Не помешают скиллы в Apache JMeter, плагинов и инструментов с GUI;
* Мобильное тестирование — нюансы тестирования нативных и гибридных приложений, мобильных интерфейсов. Сходства, различия и инструменты тестирования в iOS и Android. Автоматизация мобильного тестирования. Особенности тестирования облачных и стандартных приложений на разных девайсах;
* Тестирование доступности — специфические знания в области тестирования приложений и другого софта на доступность людям с ограниченными возможностями. В разных странах используются свои стандарты, например, в США ориентируются на протоколы 508 Accessibility Standards или WCAG.
* Тестирование безопасности — оценка ограниченности доступа к личным данным и файлам, криптография, понятия об OWASP Top 10. Знания специальной терминологии (вектор атаки, поверхность атаки), тестирование на проникновение в систему.

       Дорожная карта тестировщика на этом не заканчивается. В ходе движения по карьерной лестнице, Вы также столкнетесь с пониманием непрерывной интеграции, доставки и развертывания (CI/CD), а также множеством технологий, ведь цифровой мир развивается непрерывно. В последние годы особенно активно выделяется сфера искусственного интеллекта и машинного обучения, в которой, надо признать, роль самого тестировщика резко меняется, отличаются и обязанности других специалистов.

С сайта https://testpro.io/ru/roadmap-tester/