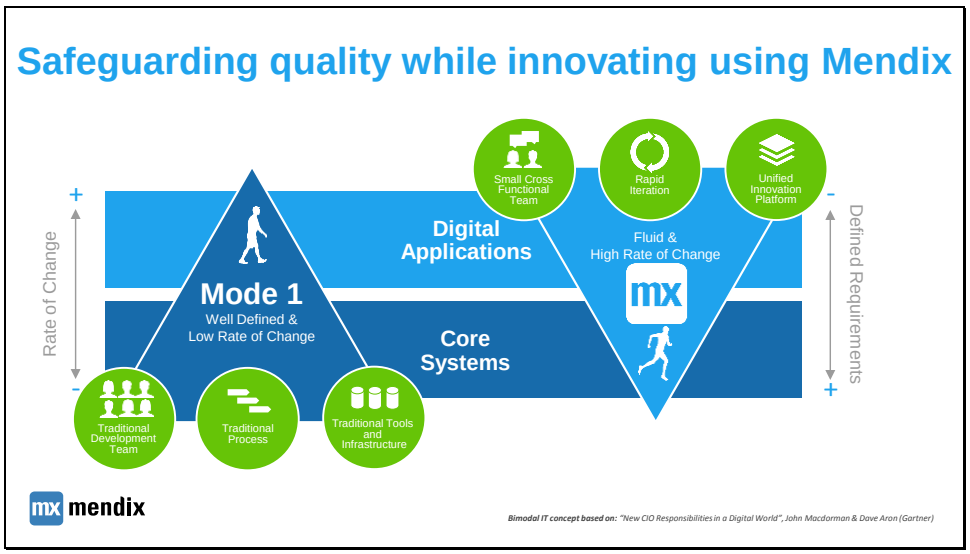
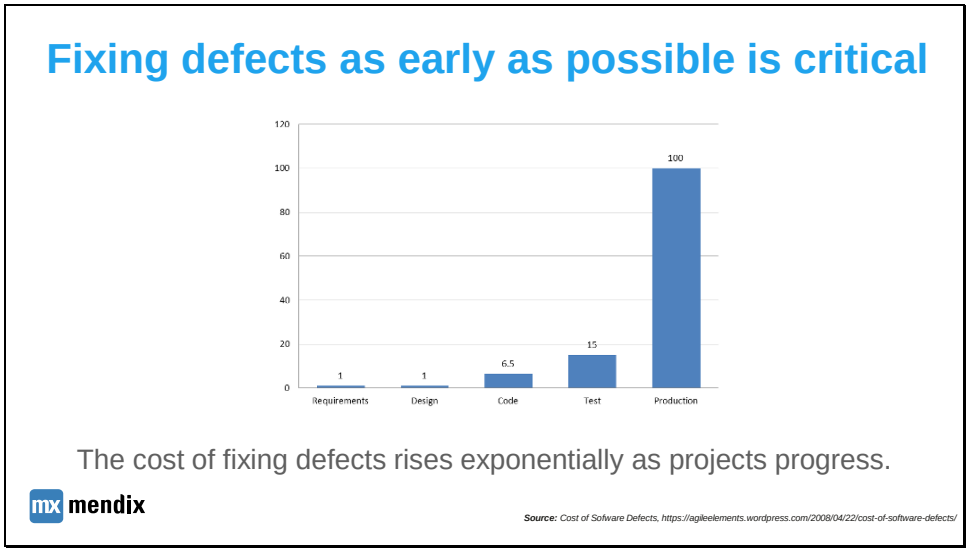
Agenda

1. Why is quality assurance important?
2. How should quality assurance be done in a Mendix context?
3. Demo of automated testing using Selenium & TestNG

mx mendix



- Gartner has identified two modes of IT delivery:
    - Mode 1: focused on stability (sequential, emphasizing safety and accuracy)
    - Mode 2: focused on agility (exploratory, emphasizing agility and speed)
- Challenge: in mode 2 you want to do quality assurance at the speed of ideas. How to do it?
- But let's first focus on why it is important to have attention for quality control in software development.



- Finding defects later is more expensive. There is a well-known saying in the software development industry that every consecutive development phase raises the cost of fixing a defect by a factor 10.
- We want to find defects as early as possible to minimize costs.

Quality Assurance Mindset in Mendix

- Quality assurance vs agility.
- Feedback is at the heart of **both** agility and quality assurance.
- QA practices are just aligned differently to maintain agility.

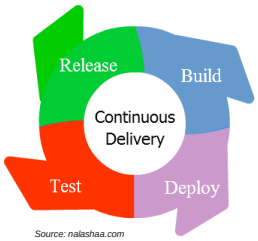
Source: studentenwerk.nl

mx mendix

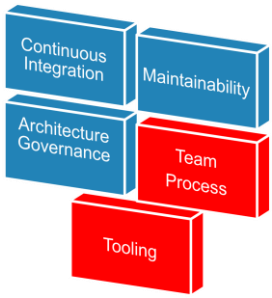
- Quality assurance and agility are at odds is a notion we hear in the marketplace.
- However, feedback is at the heart of **both** agility and quality assurance:
    - Agile emphasizes fail fast & fail often by gathering early feedback
    - While the main mechanism for quality control is also feedback on quality.
- They aspire to the same, but QA practices are just aligned differently to maintain agility:
    - Timing is different: integrated in development
    - Team members are engaged differently: jointly responsible
    - Tooling and automation: required to keep it manageable

QA is an organization-wide concern

- It entails many domains of software delivery
- Focus on process & tooling at team level
- Activities like architecture governance & continuous integration are vital



Source: nalashaa.com



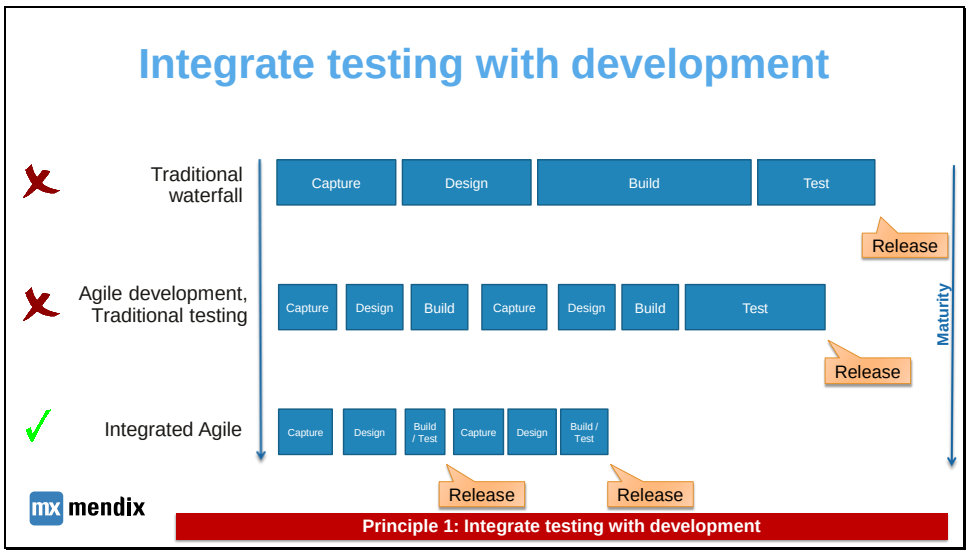
mx mendix

- Quality Assurance entails many domains of software delivery: It should be a focus point throughout your whole IT organization. Especially as you start to execute multiple projects (in parallel).
- Today we limit our focus to process & tooling on a team level: how to behave as a team towards quality software.
- In larger environments, activities like architecture governance & continuous integration are vital to continuous delivery of high-quality software. These topics are not in scope for this webinar, but need attention to ensure continued success in those contexts.

Principles

1) Integrate testing with development
2) Targeted testing
3) End-user involvement
4) Lightweight QA practices

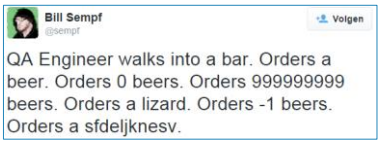
mx mendix



- Overarching goal is to shorten feedback loops.
- Ideal state is the lowest flow which depicts integrated testing & development:
    - Testing executed simultaneously with development during the sprint
    - Testing is an all-team responsibility
    - All team members are familiar with Mendix.
- The first flow depicts traditional waterfall, which has the following drawbacks:
    - Feedback comes too late, potentially triggering expensive rework
    - Increased likeliness of friction between people who focus (predominantly) on developing new features and those who focus more on testing.
- In the field we often see customers getting stuck in the middle. Agile development with waterfall testing. This approach puts your agility at risk as feedback comes in too late to handle efficiently.

Test high-risk functionality

- **Main principle:** test practices should match project risk profile.
- Use Selective testing.
- Elicit risks early and as a team effort.
- Do not test the Mendix framework itself. Mendix components are tested for you.



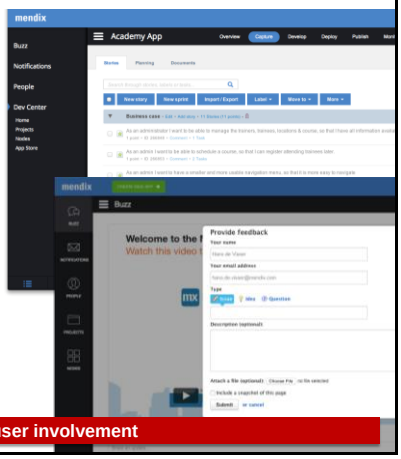
mendix

Principle 2: Targeted testing

- Do not execute all types of tests in all projects or test all functionality: test to mitigate actual business risks
- Elicit risks as a team effort, for example by doing risk poker in the Sprint Planning meeting. This ensured shared understanding.
- Do not test Mendix. See the blog accompanying this webinar for detailed information why.

Involve your end-users in upholding quality

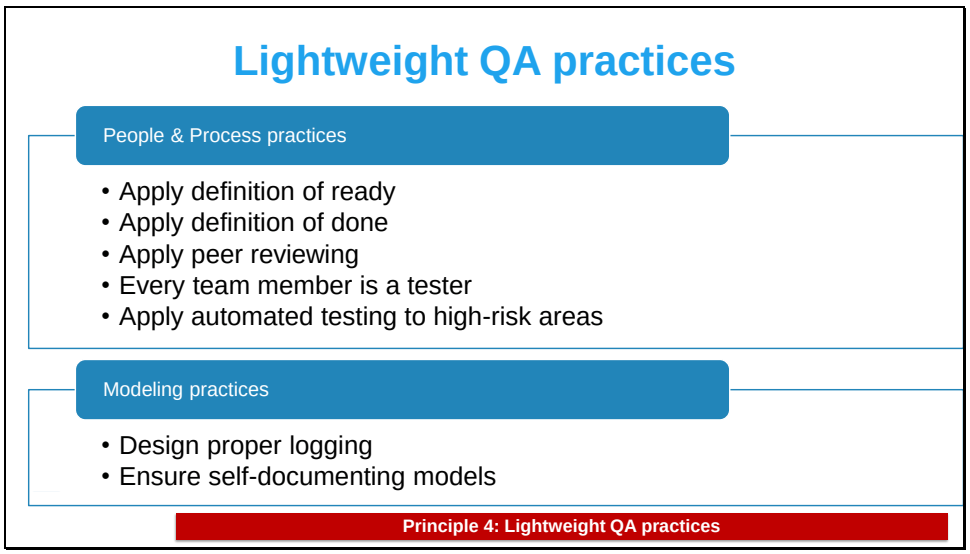
- Engage end users as early as possible
- Expose the app during development
- Use show-and-tell, working demo, beta versions
- Use one-click deployment and the feedback widget



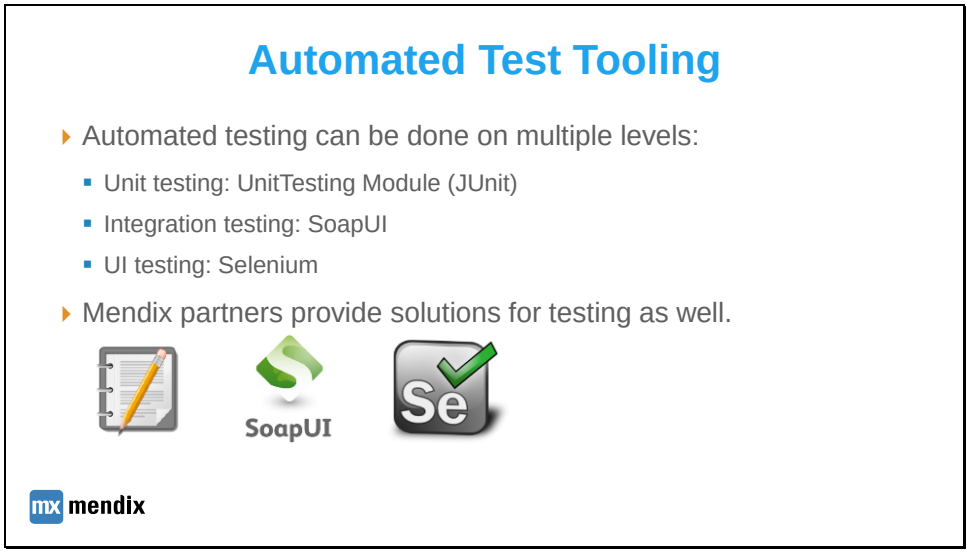
mendix

Principle 3: End-user involvement

- Engage your actual end users as early as possible: we want rapid feedback.
- Use active engagement of your users (actually logging into the system) and providing you feedback
- Mendix facilitates this rapid feedback process by offering one-click deployment & the feedback widget



Refer to the related blog about 8 lightweight Quality Assurance Practices for a detailed explanation.



As new functionality is added to the application, there is an increasing amount of existing functionality which needs to be tested. At some point automation of this regression testing is the only way to keep the release speed of your team on par.