Software Engineering 2.0.0

SELF-REVIEW 2022-03-26

Survey Summary

23 / 30	**Quality and Health**
9 / 15	**Programming Concepts**
16 / 24	**Tooling**
14 / 21	**Integration**
11 / 15	**Full Stack**
11 / 15	**Vision and Growth**
20 / 30	**Architecture**
104 / 150	**Total**

Quality and Health

Q1

K

Understands potential attack vectors within applications.

2 Familiar with the entire OWASP Top 10 for multiple application types.

References: OWASP • OWASP: Top 10 • OWASP: API Security

Q2

S

Writes code that protects against potential attack vectors.

1 Can use a library to protect against common attacks from OWASP such as XSS, CSRF, and SQL injection.

References: OWASP • OWASP: Top 10 • OWASP: API Security

Q3

B

Analyzes the team's applications attack surface and potential vulnerabilities.

2 Reviews both new code and old code for vulnerabilities and also actively refactors old code to make it more secure and resistant to attacks. Responds to security issues quickly.

References: OWASP • OWASP: Attacks • CVE

Q4

S

Understands the monitoring metrics and tools used by the company and how to integrate with them (e.g. logs, usage, system, security).

2 Can do complete setup for an application to connect to core services - either directly or through SDKs provided by the company.

Q5

S

Profiles an application to identify and implement performance enhancements.

2 Takes an active and consistent interest in profiling and monitoring and often uses tools to identify and improve expensive code paths.

Q6

K

Understands different metrics of application health and performance.

3 Understands the most important metrics for measuring application health and performance such as response time, throughput, error rates, logging and how to respond appropriately at various thresholds. Does what it takes to ensure a high-performance application (proactively and reactively).

Q7

B

Follows established development standards (security, performance, testing, coding style).

3 Sets an example for others in following standards and doing the right thing. Addresses NFRs and actively works to ensure compliance with company standards. Works hard to adhere to standards without cutting corners to ship sooner.

Q8

B

Analyzes the team's applications and fixes errors.

3 Proactively writes defensive code. Can quickly spot bugs or problematic code during code reviews. Has a deep understanding of the team's codebases and understands the importance of fixing bugs and minimizing Known Shippables.

Q9

S

Identifies, explains, and resolves software issues.

3 Can quickly troubleshoot issues, find bugs and their solutions. Efficiently identifies issues in pressure situations such as outages. The Go-To person when other developers cannot isolate and

identify issues.

Q10

S

Writes code that fails gracefully and handles error conditions (e.g. service failures, logging).

2 Can write robust code that defends against common error scenarios such as bad responses from services and log the appropriate information to allow quick and efficient troubleshooting. Understands the severity of different types of errors and when they should be suppressed or surfaced to the user.

Programming Concepts

C1

S

Writes maintainable and modular code.

3 Has written large-scale applications or libraries while maintaining solid software architecture using best practices. Writes thoughtful code that takes users into account and considers readability and comprehension a priority (principle of least surprise).

C2

K

Understands common development principles (e.g. DRY, KISS, YAGNI).

3 Understands many development principles and philosophies, how they affect the process of software development and software engineering and the quality of software - including when they are important to follow and when there may be reason to ignore or bypass them.

References: Wikipedia: Software Development Philosophies

C3

K

Understands object-oriented programming paradigms and patterns.

2 Understands OOP and more advanced topics such as domain driven design, composition, and the law of demeter. Familiar with SOLID, GRASP, and many other design patterns such as factories, repositories, facades, decorators, etc and has used them when developing.

C4

K

Understands functional programming paradigms and patterns.

0 Knows what FP is and the differences between FP and other paradigms such as object-oriented or procedural programming.

C5

K

Understands different methods of encryption (e.g. symmetric, asymmetric, authenticated).

1 Understands at a high level the different methods of encryption and how we use them at QL.

Tooling

T1

K

Understands the various types of testing and their importance (e.g. unit, functional, acceptance, user).

2 Understands the value of multiple types of test suites for an application or software and the different scenarios they help protect against. Knows how to write most types of tests for their team's projects.

T2

B

Writes comprehensive tests for all code.

2 Has used both TDD and BDD and has significant experience with at least one of them. Helps push the bar higher on code coverage and other testing metrics for projects they are involved in.

T3

S

Performs VCS management such as committing, merging, handling conflicts.

2 Can perform complex functions such as rebasing, bisect, searching through vcs history, check-in locks, stashing, and partial staging. Knows the dangers of some commands and how to avoid losing data.

T4

K

Understands best practices for multi-person or team workflows and branch/merging strategies.

2 Has an in-depth understanding of multiple development workflows such as git flow, trunk-based development, branch-based development and why some workflows may work better for certain situations. Involved in defining the workflow and improving the development and vcs integration process for their projects and team.

T5

K

Understands requirements and process for compiling and building applications in their domain.

2 Has an in-depth understanding of the processes run in order to build and compile their team's projects including system requirements and dependencies. Has experience with writing scripts for automating the build process and is keenly aware of the differences between compiling on a dev machine and for deployment. Aware of specific QL processes or caveats for our enterprise environment including consideration of our staging environments and network conditions. Can quickly debug issues that arise anywhere in the toolchain.

T6

K

Understands requirements and process for deploying applications in their domain.

2 Has an in-depth understanding of the processes run in order to deploy their team's projects to multiple environments - including both on-premises and cloud. Aware of specific QL processes or caveats for our enterprise environment including consideration of our staging environments and network conditions. Can quickly debug issues that arise anywhere in the process.

T7

K

Understands the release management pipeline and how to set up for applications in their domain.

2 Knows the entire software development process and SDLC from task intake to production deployment including requirements gathering, testing and promotion through environments. Helps define the standards and processes that the team follows for release management.

References: Wikipedia: Release engineering

T8

K

Understands the implications of changing or updating dependencies.

2 Understands the implications of using third party code and evaluates when to use them based on security, reliability, and community support. Knows how to use native package managers for their team's applications such as nuget, composer, npm, hex, and others. Has knowledge of vendoring, locked dependencies or shrinkwrapping, and verifying trust such as signing code and authorship.

Integration

I1

K

Understands the value of frameworks/libraries and how to use them effectively.

2 Digs deeply into framework and library documentation or code and knows the frameworks and libraries used by the team's applications in great detail.

I2

K

Understands how to properly implement authentication and authorization.

2 Has experience with implementing both authorization and authentication for their team's applications. Understands the security requirements and concerns for gating access to sensitive data and follows best practices both from QL and the industry.

References: RBAC • ACL • Wikipedia: OpenID • Wikipedia: SAML

I3

B

Creates and publishes reusable libraries or modules.

2 A primary contributor to several shared libraries such as service SDKs or frameworks. Encourages others to move reusable code into separate libraries for greater impact and sharing.

I4

B

Evangelizes and exhibits open-source mindset.

1 Evangelizes collaboration and open source culture.

References: ThoughtWorks: The Culture of Open Source

I5

K

Understands inter-system communication protocols (e.g. REST, SOAP, RPC, Queues).

2 Can build a service used by other services or systems using best practices and industry-standard protocols. Understands the value of following standards in designing public APIs.

References: Apache Thrift • Amazon Ion • Transit

I6

B

Respects how code changes may affect downstream consumers.

3 Takes care to present a thoughtful developer or client experience - such as simple APIs, pleasant documentation, and clear error messages. Has experience with or uses concepts such as API versioning, feature flags, semantic versioning in their systems or libraries to lessen change impact to consumers.

I7

B

Helps design and encourage community standards (e.g. performance, testing, coding style).

2 Understands the value of common standards and guides discussions and driving consensus among their team or multiple teams.

Full Stack

F1

S

Writes different types of applications (e.g. console, web, API, native, embedded).

3 Has architected or built a significant portion of several production-quality systems from multiple categories of application.

F2

K

Knowledgeable of multiple data management systems and when they are most effective.

2 Has great knowledge of at least one database system and used it extensively. Is familiar with other types of database systems. Has experience managing databases such as defining schema and writing queries, migrations, and backups.

F3

S

Can set up a complete local environment for a system (e.g. IDE, runtime, web servers, database).

2 Can set up development environment for all layers of the team's applications with minimal or no oversight.

F4

K

Understands underlying operating system and how to integrate and configure runtime for application code.

1 Understands how to install runtimes (VM, interpreter, etc) used by team's application code and run both code and tests against the runtime. Knows how to pull information out of the runtime such as version information, installed extensions or modules, etc.

F5

S

Can make changes to all levels of the team's projects (e.g. client-side, frontend, backend, database).

3 Writes high quality, idiomatic and tested code for multiple layers of the team's applications such as client-side javascript, backend .net, and database queries and schema. Has in-depth understanding of each layer and can dive deep when necessary.

Vision and Growth

V1

S

Learns new libraries and how to use them effectively.

3 Can implement new libraries and frameworks and share that knowledge with others. Can dive deep into others' code to seek a greater understanding beyond what the documentation may provide. Uses third-party code wisely and avoids being tightly coupled to specific implementations. Knows how to analyze libraries for code quality and security.

V2

B

Avoids writing code unnecessarily by using existing internal or open source solutions.

2 Seeks out new libraries and technologies in their application space and how they make their applications better. Implements third party libraries when a high quality option exists and knows when it is valuable to write their own implementation that meets the needs of QL.

V3

S

Learns new languages or systems and knows when it is appropriate to introduce them.

2 Demonstrates curiosity and interest in learning new technology and applying it to QL to make themselves and others more efficient or produce higher quality software. Works with architects

to investigate new technology or products when introduced into the company or the team's applications.

V4

B

Is actively involved with development communities (e.g. giving talks, helping others).

2 Gives talks during Hack Week and IT communities. Actively looks for ways to share knowledge and improve developer skills. Mentors other engineers in both an unofficial and official capacity.

V5

B

Stays current on new and upcoming technologies.

2 Shows active interest and curiosity in new initiatives and services or systems within QL. Stays up to date on new libraries or technology in their domain. Knows about upcoming changes or versions of libraries or frameworks used by the team and shares that knowledge with others.

Architecture

A1

B

Designs extensible and composable systems (e.g. services, APIs, modules).

2 Significant contributor to a composable software system and writes decoupled code to avoid service and vendor lock-in. Thinks ahead and considers long-term vision of their applications to minimize technical debt and increase velocity of delivery to stakeholders.

A2

K

Understands how their team's applications integrate with external systems.

2 Familiar with the dependency graph of their team's applications and what systems they are reliant on - including specific data contracts and specifications in addition to endpoints or services used. Understands service failures or errors and how to handle them gracefully with methods such as back-offs, exponential timeouts and when to surface those error conditions to consumers.

A3

K

Understands system architectural patterns and when to use them effectively.

2 Understands the benefits and trade-offs of different architectures and has experience using that knowledge to be a significant contributor to the design and implementation of a production system.

References: Wikipedia: Software Architecture Styles and Patterns

Understands the flow of data used by their team's systems.

A4

K

2 Has in-depth knowledge of methods of safeguarding data and follows best practices with regards to encryption and security. Understands the requirements and best practices for protecting data both internal and externally-accessible as well as at-rest and in-flight. Follows NFRs for data management.

A5

B

Addresses current and prevents future technical debt.

2 Writes well-thought out code that has minimal negative impact to the long-term technical wellbeing of the project. Evaluates their own code for technical debt and seeks thoughts and advice from others in the form of legitimate code reviews and design sessions. Identifies technical debt within their team's application. Frequently tackles technical debt during iterations.

References: Wikipedia: Technical debt

A6

S

Can effectively explain architectural decisions to technical and non-technical partners.

3 Can tailor messaging, documentation and presentations to contain the right amount of details for specific target audiences - including engineers and senior leaders. Thoughtfully considers feedback and input from all partners and collaborates with others to come to the best solution for the company.

A7

B

Writes effective documentation both in and outside of code (e.g. comments, changelog, architecture, system design, gliffies).

2 Writes technical and non-technical documentation both in code and outside of code such as system references, project decisions and architecture. Keeps documentation up-to-date to ensure correctness and clarity.

A8

K

Has expert knowledge in a specialized domain.

2 Actively pursues their interest as a specialist in one or more significant business-impacting domains. Shows consistent progress and growth with their experience and knowledge.

References:

Wikipedia: Domain engineering • Wikipedia: Domain knowledge • Wikipedia: Domain

• Wikipedia: Subject-matter expert • ArtSoft: Domains of expertise

---

A9

K

Understands how to create scalable software.

2 Has in-depth knowledge of design patterns to improve scalability such as statelessness, caching, lazy loading, asynchronous processing and uses them in their software designs and implementations. Works with Ops to improve handling of availability, backups, monitoring, logging, and documentation.

A10

K

Understands how to create and manage distributed software and systems.

1 Has familiarity with distributed computing paradigms and principles such as SOA, message queues, clustering, CAP Theorem, shared-nothing, and parallelism vs concurrency.