Software Engineering 2.0.0

! Survey not yet saved

Survey Summary

22 / 30	Quality and Health
10 / 15	Programming Concepts
18 / 24	Tooling
14 / 21	Integration
9 / 15	Full Stack
10 / 15	Vision and Growth
16 / 30	Architecture
99 / 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.

2 Writes code to defend against attacks when not provided by a library - and understands the security guarantees frameworks do and do not provide.

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.

1 Can connect a profiler to their application and understands its output to identify inefficient codeblocks.

Q6
K

Understands different metrics of application health and performance.

2 Knows what should be optimized within the team's applications and when optimization is less necessary due to diminishing returns.

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.

IDK I don't know.

C5
K

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

2 Understands the value of encryption and the importance of the various methods including symmetric, asymmetric, and hashing including when they should be used and what type of data they should be used with.

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.

3 Understands the meaning behind VCS commands and how they work. Comfortable with console or CLI. Can handle conflict resolution for any situation and recovering data when something goes wrong. Often helps others when they have a vcs-related issue.

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.

3 Has in-depth knowledge and experience with package managers and dependency management for multiple application ecosystems or languages. Understands the impact of relying on public systems or services for dependency management and what can be done to reduce risk of downtime for QL. Understands software licensing - both open source and proprietary - and their ramifications for software use.

Integration

I1

K

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

1 Knows the value of shared frameworks. Has used frameworks and knows how take advantage of their features to be more efficient to reduce boilerplate code and development time.

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.

2 Understands the benefits QL receives from open-source and value of contributing back to the industry. Encourages collaboration through code reviews, knowledge share, and open collaboration on software projects.

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.

IDK I don't know.

F5

S

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

2 Can make changes to all layers of the team's applications and actively improves their knowledge and familiarity with all components of the team's systems.

Vision and Growth

V1

S

Learns new libraries and how to use them effectively.

2 Can implement a new library or framework in a safe and maintainable manner. Knows where to find high quality libraries and when to use them in the enterprise.

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.

3 Actively investigates and reviews new technology such as programming languages or services and tools such as database technologies. Can justify their use and when to make use of them within QL in an enterprise setting. Understands the ramifications of introducing a new technology within QL in a mission critical system. Has discovered, researched, and implemented a new technology within QL while satisfying NFRs and company standards for quality and security.

V4

B

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

1 Participates in IT communities and actively involved in developing standards and best practices across teams and applications.

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).

3 Leads development for one or more large-scale extensible and modular systems. Drives best practices and engagement within their developer community for building better systems within the enterprise.

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

A4

K

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

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.

1 Can explain their technical decisions and implementations to other engineers and clarify software concepts to non-engineer members of their team.

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.

1 Shows an active and consistent curiosity in a specialized domain or area of expertise relevant to the company beyond their primary language. Actively looks for opportunities to be involved in company projects and initiatives in those domains.

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.

1 Familiar with metrics and tests used for determining scalability of software. Understands best practices to improve scalability of the languages and systems used by their team.

A10

K

Understands how to create and manage distributed software and systems.

IDK I don't know.