

Junior Programmer

A junior programmer assists in the development of interactive applications and experiences using real-time graphics technology. They work collaboratively with senior programmers, artists, quality assurance (QA) team members, and designers to implement game mechanics, debug and optimize code, and contribute to the overall development process. Their responsibilities may include scripting interactive behaviors, solving technical issues, and continuously learning and adapting to the evolving landscape of games and other creative industries.

Top three responsibilities



Coding and development

Write, modify, and maintain code for real-time 3D (RT3D) applications based on specifications and best practices with guidance from senior programmers.



Bug fixing and troubleshooting

Help identify and fix bugs or issues in the codebase.



Documentation and code maintenance

Contribute to documenting code and maintain clear and organized documentation for the project.

Top three skills



Coding proficiency in at least one scripting language

Demonstrate the ability to use various APIs, use consistent code styles throughout your work, and follow general coding best practices.



Ability to interpret existing code

Review and understand code bases and expand on existing code while adhering to project standards.



Debug and troubleshoot code

Diagnose and fix code that doesn't compile or isn't performing as expected.

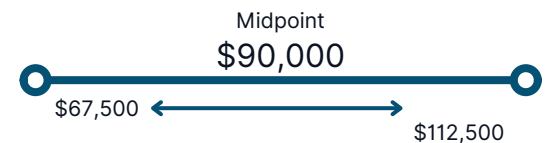
Career Stage

Entry level

0-2 years of professional experience



Pay Range



Note: These ranges are based on the Radford compensation database's 50th percentile for all industries, company sizes, sectors, and US locations. The range minimum is 25% below the midpoint, and the maximum is 25% above it. These figures are not reflective of Unity's pay ranges but provide broad benchmarks across US geographies and industries. This information should not be used to represent Unity's compensation ranges or philosophy.

Alternative Titles

A junior programmer may also have the following alternative titles:

- Junior interactive developer
- Junior software developer
- Junior software engineer

Note: Some companies might use general titles like "Programmer". Review job descriptions carefully to ensure alignment with entry-level qualifications.

Other Terms

Roles are often specified by incorporating the main tool or task into the title, such as:

- Junior Unity programmer
- Junior programmer - Unreal Engine
- Junior programmer - front end

Table of contents

About the role

[Key traits and qualities of a Junior Programmer](#)

[Responsibilities](#)

[Skills required](#)

[Tools used](#)

[Collaborative roles](#)

[Job progression](#)

Resources for career development

[Learning experiences](#)

[Key terms](#)

[Internships](#)

[Full industry list](#)

Applying for the job

[Application requirements](#)

[Resume requirements](#)

[About Applicant Tracking Systems \(ATS\).](#)

[Sample resume](#)

[Cover letters](#)

[Linkedin Profile](#)

[Portfolio requirements](#)

[Portfolio recommendations](#)

[Portfolio maintenance](#)

[General application tips](#)

[Job boards](#)



Interviewing for the job

[Interview process](#)

[Preparing for an interview](#)

[Navigating job rejection](#)

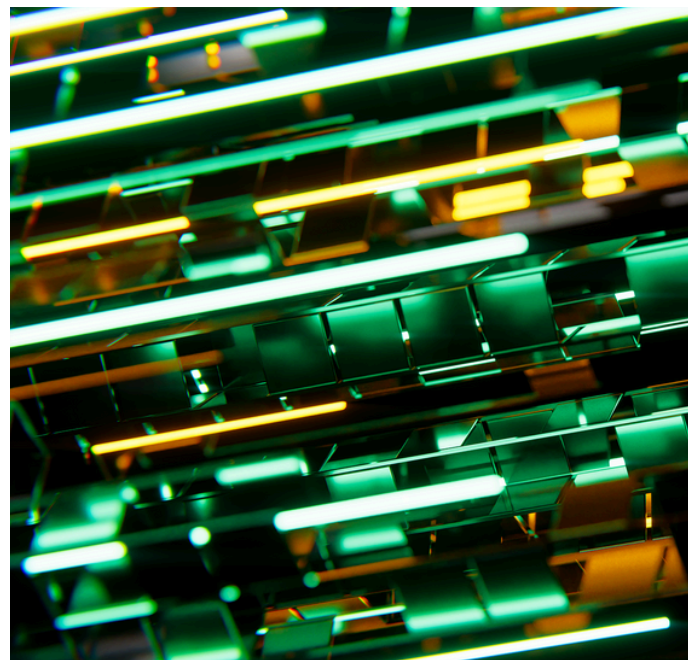
UJP information & contributions

[Acknowledgements](#)

[About the Universal Job Profiles](#)

[Contributing to the Universal Job Profiles](#)

[Changelog](#)



Job details

Key traits and qualities of a junior programmer

A junior programmer in the early stages of their career needs to balance two roles: contributing to projects and learning from more experienced colleagues. Employers value candidates who can integrate smoothly into a team, prioritizing collaboration over individualism.

While technical skills matter, clear communication and teamwork are often more important. Employers seek junior programmers who can articulate problems, propose solutions, and explain their reasoning, while also being open to feedback and different perspectives. Success in this role requires combining coding proficiency with strong interpersonal skills to thrive in a collaborative environment.


Responsibilities

When junior programmers are first getting started, their tasks will likely remain closely monitored by senior programmers and team leads. The parts of a project they will contribute to will remain small, but grow as they increase their skills and gain a deeper understanding of the project. Below are a sample of the most common responsibilities that junior programmers are assigned, but specific responsibilities will vary from company to company.



Core Responsibilities

Most junior programmers will be assigned these responsibilities. All junior programmers should be able to perform the following tasks:



Coding and development: Junior programmers are responsible for writing, modifying, and maintaining code for real-time 3D applications. They implement mechanics, user interfaces, or other interactive features based on design specifications and established programming best practices with guidance from senior programmers.

Testing and quality assurance: Junior programmers may be involved in testing the functionality and performance of the developed applications. They conduct unit testing, assist in writing automated testing scripts, assist in user testing, and contribute to ensuring the quality and stability of the software.


Bug fixing and troubleshooting: Junior programmers help identify and fix bugs or issues in the codebase. They work closely with the development team to investigate and resolve problems, whether they're related to functionality, performance, or compatibility with different platforms or devices.

Collaboration and communication: Junior programmers collaborate with other team members, such as senior programmers, artists, QA testers, and designers, to understand project requirements, estimate task duration, and contribute to the development process. Effective communication, both written and verbal, is essential to coordinate tasks, share progress updates, reach milestones, and seek assistance when needed.

Documentation and Code Maintenance: Junior programmers contribute to documenting their code and maintaining clear and organized project documentation. They also use version control systems like Git to manage code changes and collaborate effectively with the team. These practices enhance code readability, streamline collaboration, and support knowledge transfer within the team.

Secondary Responsibilities

These tasks are slightly more specialized, but it will greatly benefit junior programmers to be familiar with these tasks should they be assigned.




UI/UX consistency: Junior programmers may contribute to the establishment and maintenance of UI/UX standards across products and experiences within the company's portfolio. To do so, they collaborate closely with design and development teams to ensure consistent and user-friendly interfaces across projects.

Back end coding and development: With guidance from senior programmers, junior programmers may design, modify, and maintain the underlying systems that drive the application's functionality. This may include optimizing database queries, implementing APIs, or ensuring seamless data flow.

API development: Junior programmers may be responsible for aiding in the creation of and maintaining APIs that act as bridges between the front end and back end of an application. These APIs facilitate communication and data exchange, enabling various components to work together, from user interactions on the interface to data processing happening behind the scenes.

Personal Responsibilities

Beyond day-to-day responsibilities, junior programmers should remain focused on continuing to build their skills and knowledge base so that they can eventually progress to more senior level roles.



Ongoing learning and development: In order to grow in their chosen field, junior programmers should maintain a skills growth mindset, even once employed. They should actively invest time in learning and implementing new programming languages, frameworks, and stay up to date with the latest advances in RT3D.

Required skills

The specific tasks assigned to a junior programmer can vary significantly depending on the project they're working on. A junior programmer in games may develop aspects of a combat system, while a junior programmer working in automotive might contribute to a driving simulation. The skills listed below are generalized to be universally relevant no matter the project. These skills ensure that a junior programmer is well rounded and can adapt themselves to any job.

Script comprehension and integration:

- Ability to interpret existing code within a code base
- Experience using the features of an integrated development environment (IDE) to code efficiently and correctly.
- Ability to program scripts that integrate into existing systems.
- Ability to execute coding standards as established by senior programmers.

Effective scripting practices:

- Experience composing scripts that utilize various APIs.
- A coding style that is efficient and easy to read.
- Familiarity with coding best practices to maximize code readability and efficiency.
- Experience refactoring code for optimization and readability.
- Experience participating in code review cycles.

Data handling and persistence:

- Ability to write script functionality for data persistence within and between runtime sessions.
- Knowledge of appropriate data types and structures based on situational needs.





Bug fixing and troubleshooting:

- Familiarity with unit testing to ensure that code functions as intended.
- Ability to diagnose and fix code that compiles but fails to perform as expected.
- Ability to troubleshoot runtime exceptions.
- Familiarity with debugging applications on multiple platforms.
- Ability to profile and debug trivial performance issues.

Teamwork skills:

- Ability to give and receive feedback in a positive and helpful way.
- Ability to work in a team.
- Empathy with others in order to foster trust and respect.
- Ability to resolve conflicts diplomatically.

Technical proficiency:

- Excellent writing skills for documentation and internal communications.
- Familiarity with common project management methods, such as agile and waterfall
- Familiarity with Key Performance Indicators (KPIs) and Objectives and Key Results (OKRs)
- Familiarity with popular task management software, such as Jira or Trello
- Knowledge of fundamental version control concepts.

Personal development:

- Ability to manage your time well to balance work, personal life, and relaxation for a healthy lifestyle
- Habits for handling stress, such as mindfulness practices, to cope with the busy game industry.
- A growing professional network in the game industry, cultivated by joining forums, attending conferences, and going to meetups

Tools used

In games and broader creative industries, junior programmers make use of a large variety of tools that are specially designed for specific tasks. The following list highlights important tool categories that are often required for day-to-day work. Junior programmers should be proficient in at least one tool from each category. Demonstrating an understanding of how and why a category of tools is used is more important than knowing the specific programs a company uses.


Integrated development environments (IDEs): IDEs such as **Xcode**, **Android Studio**, **Visual Studio**, or **JetBrains Rider** provide a comprehensive coding environment with features like code editing, debugging, version control integration, and project management.

Real-time 3D engines: Real-time 3D engines like **Unity** or **Unreal Engine** are widely used in the industry. These engines provide a suite of tools for designing, building, and deploying interactive applications, including game development, simulations, and virtual experiences.

Programming languages: Languages suited for real-time 3D development include **C#**, **C++**, **Lua**, and **Python**, among many others. The choice of language will depend on the specific engine, project requirements, or company preferences.

Version control systems: Version control systems like **Git** or **SVN** are crucial for collaboration and code management. These tools are used to track changes, manage code branches, and collaborate with other team members.





Debugging and profiling tools: Debugging and profiling tools, such as **Visual Studio Debugger** or **Unity Profiler**, help in identifying and troubleshooting issues in code. These tools assist in understanding runtime behavior, performance bottlenecks, and memory management.

Project management and communication tools: Project management tools like **Jira**, **Trello**, or **Asana** are used to track tasks, collaborate with team members, and communicate project updates. Communication tools like **Slack** or **Microsoft Teams** facilitate real-time communication and collaboration within the development team.

Artificial intelligence (AI) tools: While still very new, AI tools such as **ChatGPT**, **TensorFlow**, or **PyTorch** are already being integrated into production workflows to assist with a wide array of programming tasks. These tools are able to generate code snippets, provide code completion suggestions, and offer solutions to coding issues.

Cloud services: For junior programmers focusing on back-end development, platforms such as **AWS** (Amazon Web Services), **Azure**, **Unity DevOps**, and others offer scalable infrastructure and services for hosting applications, databases, and server-side components.



Collaborative roles

Junior programmers typically work closely with several departments on a day-to-day basis, collaborating as part of a larger development team. The following list includes common job roles that junior programmers may work with:


Senior programmers/developers: Junior programmers work closely with senior programmers or developers who provide guidance, mentorship, and oversight. They collaborate on coding tasks, share knowledge, and seek guidance on more complex programming challenges.

Artists and designers: In real-time 3D industries, collaboration with artists and designers is crucial. Junior programmers work closely with these professionals to implement their visual assets, integrate animations, optimize performance, and ensure that the interactive elements align with the artistic vision and design specifications.

Technical artists: Technical artists bridge the gap between art and programming. They help implement and optimize art assets, create shaders, set up visual effects, and ensure the technical feasibility of the artistic vision. Junior programmers may collaborate with technical artists to incorporate their work into the overall development process.

Producers and project managers: Producers and project managers oversee the development process and ensure that projects are completed on time and within budget. Junior programmers interact with them to provide progress updates, receive task assignments, and discuss project requirements or changes.





Quality assurance testers: Junior programmers work closely with QA testers to identify and fix bugs, ensure functionality, and optimize performance. They collaborate to reproduce and understand reported issues and work together to resolve them effectively.

Sound engineers: For projects involving audio, junior programmers collaborate with sound engineers to integrate sound effects, music, or voice-over assets into the application. They work together to synchronize audio cues with visual elements and create an immersive auditory experience.

Data Scientists/Analysts: Depending on the company and project, junior programmers may work with data scientists or analysts to implement tracking mechanisms, collect and analyze data, and integrate data-driven features or systems into the application.

Job progression


One of the greatest advantages of starting out as a junior programmer is the wide range of career opportunities that open up with experience. While many programmers choose to specialize in specific areas as they advance, focusing on one niche isn't the only path to career growth. There are also opportunities to branch into broader roles, such as technical art or project management. Below is a brief list of potential paths junior programmers can explore as they develop their careers:

Back-end programmer: Back-end programmers manage server-side infrastructure for RT3D applications that include multiplayer, collaborative, or otherwise dynamic experiences. They specialize in designing scalable and secure systems, managing data, and implementing network communication protocols. A junior programmer could transition into a back-end role by learning about server-side programming and architecture, database design and maintenance, and API development.

Front-end programmer: Front-end programmers play a pivotal role in shaping the user interface and interactive elements of real-time 3D applications. They focus on crafting visually engaging and responsive user experiences, incorporating assets, animations, and UI. Junior programmers can transition into front-end roles by becoming familiar with creating and/or implementing 3D assets, understanding UI design principles, and becoming proficient at creating user focused interactive experiences.

Mid-level programmer/developer: With increased experience and proficiency, junior programmers can progress to mid-level programmer roles. In these positions, they take on more complex programming tasks, have greater autonomy, and may mentor junior programmers. Junior programmers can transition into this role by contributing to larger parts of a project and by demonstrating overall growth in programming skills.





Technical artist: Technical artists bridge the gap between art and programming. They specialize in creating efficient pipelines, developing shaders, optimizing visual assets, and implementing advanced graphics techniques. Junior programmers with an artistic inclination can transition into technical artist roles by honing their skills in 3D modeling, animation, or visual effects (VFX) production, and by creating tools extensions for RT3D software.

Interaction/gameplay programmer: Interaction/gameplay programmers focus on designing and implementing interactive systems and gameplay mechanics within RT3D applications. Junior programmers with a strong interest in interactive experiences or game mechanics can move into these roles by actively participating in game development projects and studying algorithms and techniques related to interaction design.

Tools/engine programmer: While many companies work with pre-built engines and tools, some still prefer to create their own in-house solutions. Tools/engine programmers create these systems according to team needs and project requirements. Junior programmers can prepare for this role by actively engaging with and expanding on existing tools and engines, creating their own, contributing to tool improvement initiatives, or gaining proficiency in software architecture.

Technical project manager/producer: Technical project managers or producers oversee the planning, execution, and delivery of technical projects, and manage resources, timelines, and budgets. Junior programmers who become interested in project management after gaining production experience can move into these roles by collaborating with other project managers, acquiring proficiency in project management methodologies, and cultivating strong communication skills.

Systems engineer: Systems engineers are responsible for scoping, developing, and supporting core gameplay systems. They often work in the low-level side of the engine to create systems for game features, developer tooling, and optimization. This work may touch a variety of different systems including streaming, SDKs, threading, memory management, and platform and middleware abstractions. Junior programmers interested in this role should participate in engine development projects and study system architecture.

Graphics engineer: Graphics engineers create, maintain, and optimize rendering systems for video games, and they require expertise in advanced mathematics, real-time computer graphics, threading, memory management, and low-level programming. Junior programmers interested in this role can prepare by building a strong foundation in mathematics, including linear algebra, trigonometry, and calculus, exploring real-time graphics concepts, and gaining proficiency in low-level programming languages, such as C++.

Resources for career development

Learning experiences

Junior Programmer Learning Pathway on Unity Learn: This complete learning experience is designed for anyone interested in learning to code and obtaining an entry-level role using Unity. This pathway assumes a basic knowledge of Unity and has no math prerequisites. The Junior Programmer Pathway also prepares you for the Unity Associate Programmer certification.

Certifications

Unity Certified Associate Programmer: This certification validates your Unity programming skills to employers by demonstrating core skills and competencies across programming, UI, debugging, and asset management.



Key terms

The game and broader creative industries have their own set of words and phrases that might seem confusing to outsiders. Junior programmers will come across specific terms that they'll need to know in order to do their job well and work with others. To help aspiring programmers prepare and stand out as strong candidates, below is a list of important terms commonly encountered in a programming career. Learning these words and phrases will not only enhance understanding of the role, but will also provide the skills and confidence needed to succeed in this ever-changing industry.

Object-Oriented Programming (OOP): A programming paradigm based on the concept of "objects," which can contain data (attributes) and code (methods).

Key concepts include:


- **Classes:** Blueprints for creating objects.
- **Objects:** Instances of classes.
- **Inheritance:** Allows a class to inherit properties and methods from another class.
- **Encapsulation:** Bundling data and methods together, restricting access to some components.
- **Polymorphism:** The ability for objects of different classes to be treated as objects of a common superclass.
- **Design Patterns:** Reusable solutions to common programming problems or challenges. Examples include the Singleton pattern, Factory pattern, and Observer pattern. Understanding these patterns is critical for writing maintainable, scalable OOP-based code.

Functions/Methods: Reusable blocks of code that perform a specific task and can be called upon when needed.

- Note: The terms "function" and "method" are often used interchangeably. However, technically, a "method" refers to a function that is tied to an object in object-oriented programming, while a "function" can exist independently.

Static Variables and Methods:

- **Static Variables:** Variables that belong to a class rather than a specific object. Their values are shared across all instances of that class.
- **Static Methods:** Methods that belong to a class and can be called without creating an instance of that class. They typically operate on static variables or perform general utilities.



Algorithm: A step-by-step procedure or formula for solving a problem or accomplishing a task.

Data Structure: A way of organizing and storing data so that it can be accessed and modified efficiently.

Control Structures: Constructs that dictate the order in which instructions are executed. Examples include loops (for, while) and conditional statements (if-else).

Libraries and Frameworks:

- **Libraries:** Collections of pre-written code that developers can use to save time and effort.
- **Frameworks:** Provide a structured environment and predefined tools for building applications, often dictating the architecture of the project.

API (Application Programming Interface): A set of tools, definitions, and protocols that allow different software systems or components to communicate and work together.

Database: An organized collection of data, generally stored and accessed electronically.


Debugging: The process of finding and fixing defects or problems within a program to ensure it functions correctly.

- **Debugger:** A specialized tool that helps programmers investigate and fix issues in their code. It allows you to pause a program at specific points (breakpoints), inspect variables, step through code line by line, and monitor program behavior.

Version Control System (VCS): A tool or system that helps manage changes to source code. Popular tools include Git, Mercurial, and Subversion.

- Note: Version control and source control are often used interchangeably. However, source control specifically refers to managing changes to source code, whereas version control is a broader term that applies to tracking changes in any set of files.

Code Review: The practice of systematically examining code written by another developer to identify mistakes overlooked in the initial development phase, improve code quality, and facilitate knowledge sharing.



Unit Testing: A type of software testing where individual units/components of a software are tested in isolation to ensure they work as intended.

Automated Tests and Manual Tests: Automated tests rely on specialized tools to execute repetitive tasks, while manual testing involves humans testing the application by hand. Both are critical for ensuring code quality.

Agile Methodology: A set of principles for software development under which requirements and solutions evolve through collaborative effort. Common practices include Scrum and Kanban.

Game Development Version Release Terms:

- **Alpha:** An early version of a game that is still in development and typically not feature-complete. It's often used internally or shared with a limited audience for initial feedback, with many bugs expected.
- **Beta:** A more polished version of the game, but still with potential bugs or missing optimizations. It's often shared with a broader audience or testers to gather feedback before the final release.
- **Release Candidate (RC):** A near-final version of the game that is considered stable and ready for release, unless significant bugs or issues are found.
- **Gold/Final Release:** The final version of the game that is distributed to the public or sent to platforms for publishing.

Real-Time 3D (RT3D): A term used to describe three-dimensional graphics that are rendered and displayed in real time as the user interacts with them. RT3D engines (like Unity or Unreal Engine) continuously calculate and update the position, lighting, and appearance of objects in the scene, enabling dynamic and interactive experiences like video games, simulations, and virtual reality.

Internships

Though not always widely recognized, the gaming industry does provide internship programs, often hosted by larger studios. These internships deliver vital hands-on experience and serve as a gateway to entry-level positions. Industry internships are generally seasonal. Interested candidates should begin searching for openings as early as February to ensure their applications align with the recruitment timelines for summer programs. Information about internships can typically be found on company websites, and once available, these opportunities are often listed on job boards like [Hitmarker](#).

Several game studios offer regular internship programs, providing opportunities for students and recent graduates to gain industry experience. Here are a few notable ones:

[Activision Blizzard](#) - Known for franchises like Call of Duty and World of Warcraft, Activision Blizzard offers internships in game development, data analysis, and business operations.

[Electronic Arts \(EA\)](#) - EA offers a range of internships across various departments, including game development, design, and business operations.

[Epic Games](#) - The studio behind Fortnite offers internships in software engineering, game design, and more.

[Insomniac Games](#) - Creators of games like Spider-Man and Ratchet & Clank, Insomniac offers internships in various disciplines.

[Niantic](#) - Creator of augmented reality games like Pokémon GO, Niantic offers internships in fields such as software engineering, game design, data science, and user experience design.

[Riot Games](#) - Creators of League of Legends, Riot Games provides internships in areas such as game design, software engineering, and art.

[Sony Interactive Entertainment](#) - Offers internships in game development and business functions through PlayStation.

[Ubisoft](#) - With internships available in multiple countries, Ubisoft offers roles in game design, programming, art, and marketing.

Industry list

A junior programmer's skills put them in the unique position of being in demand across a wide variety of industries that use real-time 3D tools. This offers more opportunities when a junior programmer is first starting out, and excitingly, the skills that they gain in one sector transfer to others without issue. Below is a list of common industries that hire junior programmers:

- Aerospace and defense
- Animation, media, film, and entertainment
- Architecture, engineering, and construction (AEC)
- Automotive
- Education and training
- Energy and natural resources
- Games
- Healthcare
- Manufacturing and engineering
- Marketing and advertising
- Retail and ecommerce



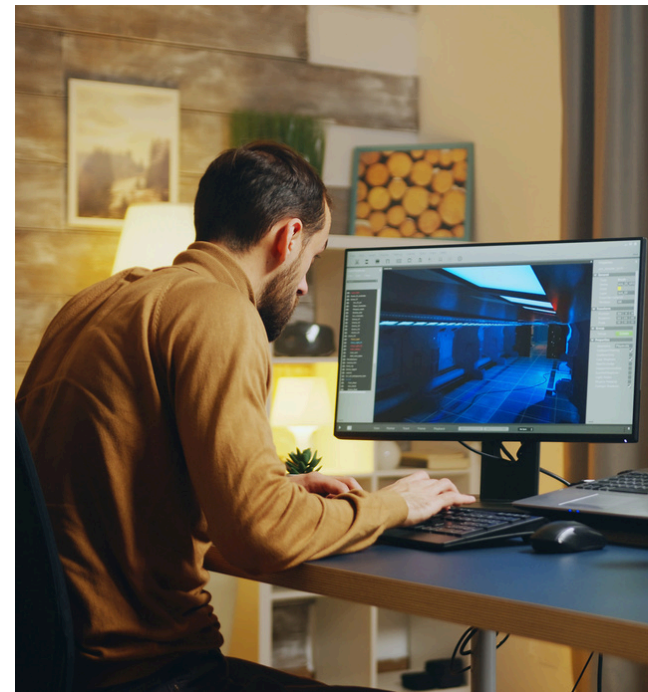
The application process

Prepare for the job hunt

Once you've developed your skills as a junior programmer, the next big step is landing the right job. Part two of this guide acts as a roadmap to help you navigate the often challenging process of job applications and interviews.

In a competitive field, success in the job market is not just about having strong coding skills—it's also about showcasing them effectively to potential employers. This section covers how to craft compelling resumes and cover letters, build an impressive portfolio that highlights your unique abilities, and optimize your LinkedIn profile to attract opportunities. It also provides practical advice on interview preparation and presenting yourself with confidence.

You'll also find strategies for sharpening your job search to target roles that align with your career goals in this section. Special emphasis is placed on developing resilience when facing rejections and learning how to use those experiences to fine-tune your approach moving forward.



Resume

A resume is a vital tool for anyone seeking employment in the RT3D industry. Even when you're starting out in the industry and have little experience to showcase, your resume is an opportunity to highlight your skills and knowledge, and also serves as a way to point employers to samples of your work. You will often be asked to provide a resume in addition to filling out information about yourself in an application. Having a resume already prepared will help save you time during your job search.

When preparing a resume, be sure to include the following information:

- Name and contact information:** This should be the full name you go by in a professional setting. If you are concerned about your contact information being publicly available, it's okay to minimize the information you include. However, you must have at least one contact method, such as an email, through which an employer can contact you to arrange an interview.
- Desired title:** This should align with the job you're applying for (in other words, Junior Programmer).
- Skills:** List your technical skills, including specific scripting languages and software packages, in bullet format.
- Projects:** Any projects you have worked on, and your specific role in them if on a team. Projects that you worked on while in a training/academic program are fine to list here. If you have any relevant work that has been published, be sure to include it.
- Links to your work:** Relevant links to your LinkedIn, portfolio, github, or other work samples
- Education:** School or other forms of training, if applicable.
- Certifications/certificates:** Anything you earned during the course of your learning for this role that is formally recognized, if applicable.
- Internships/apprenticeships:** Any formalized training experience you participated in, if applicable. Be sure to include information on the company that managed your internship/apprenticeship.
- File name:** Ensure that the file name of your resume is simple, descriptive, and most importantly contains your full first and last name.

Automated Tracking Systems (ATS)

An important aspect of resume preparation to keep in mind is that today most employers make use of applicant tracking systems (ATS), which are a type of software that help companies manage the recruitment process. An ATS automates the process of sorting and filtering resumes to help identify likely candidates for a human reviewer. While it might seem frustrating that a computer reviews your resume before a person does, this enables recruiters and hiring managers to spend more time on resumes and potential job candidates than they would be able to otherwise. Because the first step of the application process is managed by computers, it's extremely important that you format your resume so that it's optimized for an ATS.

When preparing your resume for an ATS, be sure to review:

- Keywords:** Include relevant keywords in your resume that match the job listing. ATS often scans for specific words or phrases to determine the relevance of an application. For example, if the job listing is looking for experience with Unreal Engine, and you know both Unity and Unreal, do not list "various game engines", but explicitly list the engines by name.
- Formatting:** Use a clean and simple format. Avoid complex layouts, images, or graphics that may confuse the ATS. It's a general best practice to avoid including any images, especially a photo of yourself in your resume.
- File format:** Submit your application in a format that the ATS can easily read, such as plain text or a common document format like .docx or .pdf. It's a good idea to have your resume ready in multiple formats ahead of time. Most word processing programs allow you to export to multiple formats. When uploading your resume to an application page, take special care to upload using the recommended format.
- Section headings:** Clearly label sections of your resume (for example, "Work Experience", "Education", "Skills", etc.) to help the ATS categorize information accurately. Don't use specialized terms or uncommon acronyms in headers.
- Bullet points:** Present information using bullet points for clarity. ATS systems often prefer straightforward, concise content.
- Special characters:** Minimize the use of special characters, symbols, or unusual fonts, as these may not be interpreted correctly by the ATS. Default fonts found in most word processing programs are generally a safe choice.

Sample resume

Below is an example of a resume that follows the guidelines outlined above.

Alex Ample

Programmer

(123) 456-7890 | alex@example.com | linkedin.com/in/alexample | github.com/alexample | aaportfolio.com

Education

Bachelor of Science, Computer Science

Example University, City, State

GPA: 3.8 | June 2024

- Relevant Coursework: Game Development and Design, Data Structures and Algorithms, Artificial Intelligence for Games, 3D Graphics Programming, Interactive Narrative Design
-

Technical Skills

- Programming Languages: C++, C#, Java, Python, JavaScript
 - Game Engines: Unity, Unreal Engine
 - Tools & Technologies: Git, Blender, Autodesk Maya, Visual Studio, GitHub, Jira
 - Web Technologies: HTML, CSS, React
 - Other Skills: Object-Oriented Programming, Agile Methodologies, Version Control, SQL
-

Experience

Programmer Intern

Example Game Studio, City, State | June-September 2023

- Worked closely with senior developers to implement game mechanics and features using C++ within Unreal Engine.
 - Debugged and optimized code to improve game performance and stability.
 - Assisted in developing and maintaining the game's AI logic and behavior trees.
 - Participated in code reviews and provided constructive feedback to peers.
-

Projects

Mystic Example Project | Lead Programmer | Unity, C#, Blender | [Steam Link](#)

- Developed core game mechanics including player movement, enemy AI, and collision detection.
- Implemented an inventory system and UI components using Unity's component-based architecture.
- Created custom shaders and effects to enhance visual performance.
- Conducted rigorous testing and code refactoring to ensure high performance and minimal bugs.

Space Example Project | Gameplay Programmer | Unreal Engine, C++ | [GitHub Link](#)

- Implemented dynamic shooting mechanics and enemy AI patterns.
- Developed optimized pathfinding algorithms using A* and NavMesh in Unreal Engine.
- Integrated real-time multiplayer functionalities using Unreal's networking framework.
- Created automated testing scripts to ensure consistent game behavior across builds.

Cover letters

While often considered one of the most time consuming aspects of applying for a job, cover letters are the first chance you have to introduce yourself to a company using your own words, and therefore represents an important opportunity. While an ATS may scan your cover letter for keywords much in the same way it does your resume, it's far more likely that an actual person will be reading your cover letter. It's common for people just entering the industry to create generic cover letters or even skip them entirely, so taking the time to craft a meaningful cover letter will help the reader remember you, and this may lead to an increased chance of getting an interview. Take care to make a positive and meaningful first impression.

While you may be able to reuse some content between cover letters, such as a personal introduction or an overview of your skills, most of a cover letter should be written specifically for the company you're sending it to. A cover letter should express why you would be a good candidate for the role, what specifically drew you to the job, and any interesting anecdotes or additional information that might pique the reader's interest.

A cover letter should be one page or less, and should contain the following information:

- A brief introduction of yourself
- What interests you about the company
- What made you want to apply for the role
- What makes you uniquely qualified for this specific job
- Thank the reader for their time

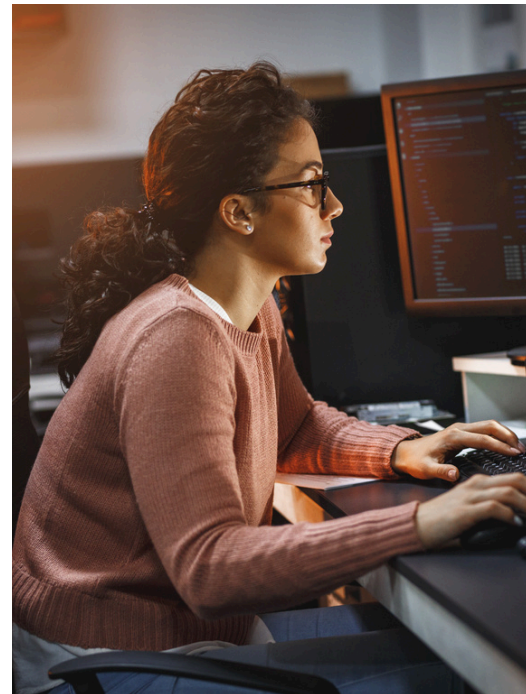


LinkedIn profile

A strong LinkedIn profile is essential in the game and creative industries, though many new job seekers underestimate its importance. Recruiters frequently use LinkedIn for candidate evaluations, and lacking a profile can raise red flags. Beyond showcasing your professional presence, LinkedIn offers opportunities to network, stay updated on industry trends, and discover job openings—often announced here first. A well-crafted profile elevates your visibility and serves as a key tool for career growth.

When creating your LinkedIn profile, consider the following:

- It is a professional space:** While LinkedIn can be considered a social media site, it's one for professional use exclusively. Use LinkedIn with the expectation that potential employers will see everything you post and include on your profile.
- Create your resume first:** Having your resume created first will significantly speed up the process of creating your LinkedIn profile.
- Customize your LinkedIn URL:** Personalize your LinkedIn URL to make it easy to share. A good rule of thumb is to make your URL your name.
- Join and participate in groups:** Join LinkedIn groups that align with your interests to connect with fellow professionals in the industry you wish to join. Engage in discussions and share your insights in a respectful, professional manner.
- Including a professional photo is normal:** Unlike on a resume, LinkedIn profiles can include a personal photo. This should be a professional, clear image of yourself, not a group shot. Essentially, choose a picture that would be suitable for a school or work ID.



Portfolio

A portfolio is one of the most important assets of any creative professional, serving as a showcase of your current capabilities in your chosen area of work. It acts as your visual resume, providing potential employers with important insight into your skills, style, and approach to problem solving. While on the job hunt, it's crucial to continually refine and improve your portfolio, ensuring it accurately reflects your improving skills. This section highlights practical details of what your portfolio should include for the application process.

When preparing your portfolio to be reviewed with your application, be sure that includes the following:


Your name and contact information: This should be included in case the hiring manager reviewing your portfolio loses track of your resume. Ensure you're easy to contact from the portfolio itself. Consider including a link to your LinkedIn profile or to your resume.

Project descriptions: Provide clear and concise descriptions for each project, explaining the goals, features, and technologies used. Highlight any unique challenges or innovative solutions you implemented. This helps prospective employers understand the scope and complexity of your work. Be sure to note if you developed a project as part of a team, and what role you performed.

Published projects: Highlight projects that have been fully published and specify the platform they are available on. Published works underscore your ability to work across the entire production pipeline, which shows a deep understanding beyond prototype creation. Published projects are significant achievements and are of particular interest to employers.

Visual assets: Incorporate visual assets such as screenshots, videos, or interactive demos to showcase the visual quality and functionality of your projects. Visual elements provide a tangible representation of your work and make it easier for employers to assess your skills.



- 
- Ease of navigation:** When putting your portfolio together, consider the type of content that you'll be showcasing and select a platform that will best serve that kind of content. If you choose to create your own custom website to host your portfolio, ensure that viewers can easily find the full contents of your portfolio with a minimum number of clicks.
-

Portfolio recommendations

The contents of a portfolio will always vary based on its creator. However, when you're just starting out, it can be challenging to come up with ideas for portfolio pieces. If you're struggling, spend time studying the games and media you enjoy the most. Ask yourself why you enjoy them and try to identify the specific systems that make them fun. Consider if there are elements within those systems that you can draw inspiration from or even recreate.


Remember, a portfolio should reflect not only your skills but also your interests and values.

As a junior programmer, always include code samples in your portfolio. Employers hiring for programming roles want to assess the quality of your work, and this can't be achieved if you only show a final project. Code samples not only reveal your technical proficiency but also provide valuable insights into your thought process and problem-solving approach.

Below are a few examples of portfolio pieces that would be appropriate for a junior programmer portfolio:

A simple complete game: If your career goal is in game development, showing your ability to program all aspects of a game is a valuable portfolio piece. Focus on a smaller-scale project, such as a simple arcade game, a card game, or an idle game. The key is to demonstrate your proficiency in creating functionality for essential game components.

A full system: For those interested in showcasing their capability to create functionality integrated into larger applications, a full system design is a great option. This system could range from a fighting system for a game to an inventory management system for a manufacturing application or a quiz system for a testing application.



Standalone code samples: Include snippets of code that show your programming skills. Focus on demonstrating clean and well-structured code, adherence to coding standards, and problem-solving approaches. Be sure to include comments in the code to provide context and explain your thought process. These code samples don't need to be part of a complete project but should demonstrate your knowledge or ability in a specified area, including how the code interacts with other systems in the application.

Depending on your specific area of interest, some other portfolio pieces might be:

User interface (UI) design and implementation: Demonstrate your proficiency in designing and implementing user interfaces. Create a portfolio piece that showcases your ability to enhance user experience through intuitive and visually appealing interfaces.

Procedural generation: Showcase your skills in procedural content generation by creating a system that generates game levels, landscapes, or other content dynamically. This can be particularly impressive for those interested in game development.

Networking and multiplayer functionality: Develop a project that demonstrates your understanding of networking and multiplayer functionality. This could be a simple multiplayer game or an application that relies on real-time collaboration.

Tool development: Design and implement a custom tool that enhances the development process. This could be a utility for asset management, a debugging tool, or any other tool that streamlines tasks for RT3D development.

Integration with external APIs: Showcase your ability to integrate external APIs into your projects. This could involve incorporating weather data, geolocation services, or other APIs relevant to your chosen project.

A note on art

Many programmers worry about including visual elements in their portfolio because they think it will distract from the code that they've written. This is an unnecessary concern. If you're building a game and are using open source assets, commissioned art, or even placeholder art (sometimes called programmer art), simply use good quality project descriptions to define what the viewer should be focusing on in the piece. Remember to appropriately credit any assets that you use.



Portfolio maintenance

A portfolio is an asset that you should regularly curate as your skills grow and evolve. It is also a very good place to focus your efforts on as you wait for new job opportunities to become available. Consider the following when maintaining your portfolio:

- Regularly remove outdated work:** Ensure your portfolio always aligns with your current skill level. Regularly review and eliminate pieces that no longer reflect your expertise or current approach to work. This ensures that viewers are able to accurately estimate your skill level.
- Avoid unedited tutorial work:** Early on, your portfolio may include tutorial or assignment pieces. Improve these by adding variation or extra content for uniqueness, making your portfolio stand out from others who used the same tutorials.
- Show your personality with your work:** Use your portfolio to showcase your interests, values, and unique style to potential employers through diverse projects that highlight your technical skills and problem-solving approach.
- Focus on quality and diversity of work:** Choose fewer, high-quality projects for your portfolio to showcase diverse skills. Each should highlight your technical abilities, problem-solving, and creativity. Include more than one example to show potential employers your skills.

The importance of portfolio specificity

When you begin your job search, it may be tempting to showcase everything you can do by including a wide variety of samples in your portfolio. For instance, if you're a programmer with an interest in character art, you might consider adding your character models alongside your code samples. However, this approach can have a negative impact on your job prospects. A well-curated portfolio should reflect the specific roles you are currently applying for. Recruiters often have very little time to spend on each portfolio they review, and need to be able to quickly understand your primary area of expertise. Presenting a wide array of skills can muddle your focus and you are likely to be judged by your weakest skill. If you insist on pursuing multiple job types, create separate dedicated portfolios for each.

Application tips

- Spell check:** Carefully check your resume, cover letters, portfolio, and LinkedIn profile for spelling errors. If possible, have your documents reviewed by another person to help identify any words that are spelled correctly, but used in the wrong context (for example, do you actually have a “Skulls” header in your resume, rather than a “Skills” header?).

- Find the hiring point of contact:** When applying for jobs, identify and connect with the hiring manager or recruiter via the company's site or LinkedIn. After applying, express your interest in the role to show proactivity. This gets you noticed, creates a good first impression, and aligns you with the goal of finding a proper fit, increasing your chances of standing out.

- Ask questions during the interview:** Have questions ready for your interview. This shows your interest in the role and helps you understand expectations and company culture. Being question-less could appear as disinterest or lack of preparation.

- Follow up:** Follow up with all communication during the application process. It shows politeness, an appreciation for people's time, and reinforces your interest. Respond to emails/calls promptly but not outside of working hours. Use follow up emails to thank people, ask additional questions, or clarify next steps post-interview.

- Assess company fit:** Remember, interviews are a two-way street. Just as the company is evaluating you, assess if you'd thrive there. Don't rush into unsuitable jobs due to circumstances, as you may end up job hunting again soon. During interviews, gauge if the company matches your values and work style for a better career fit.



Job boards

While traditional job boards can feature game industry jobs, job seekers will often have better luck using industry specific boards. These platforms concentrate on gaming-related positions ranging from development and design to quality assurance and production. These industry specific boards are invaluable tools for both emerging professionals and experienced individuals seeking new opportunities that are fine-tuned to their expertise. Below is a list of a few industry specific boards:

- [Amir Savat's Games Community](#)
- [Gamesindustry.biz jobs board](#)
- [Games Jobs Direct](#)
- [Grackle HQ](#)
- [Hitmarker](#)
- [Work With Indies](#)



The interview process

Interviews for junior programmer positions typically include more than one round of interviews. These may be a mix of behavioral interviews to assess your interpersonal skills, teamwork, and cultural fit, as well as technical interviews to evaluate your technical knowledge and problem solving approach. Technical interviews may involve discussing real-time 3D concepts, algorithms, data structures, or specific programming languages and frameworks.

Initial screening: A hiring manager or recruiter conducts an initial screening to assess your basic qualifications, interest in the role, and understanding of the target industry. This stage may involve a review of your resume and a preliminary phone or video interview.

Technical Assessment: Many companies conduct a technical assessment to evaluate your programming skills and problem-solving abilities. This may involve coding exercises, algorithmic problems, or even a take-home coding assignment. Be prepared to showcase your coding proficiency and demonstrate your ability to solve programming challenges. For take-home assignments, ensure you allocate sufficient time to complete them and submit within the specified timeline.

Coding interviews: Some interviews may include hypothetical problem-solving scenarios or coding challenges to assess your ability to communicate, think critically, and solve problems in real-time. This may involve working through a coding problem on a whiteboard (either physical or digital) or explaining your approach to a given scenario. These interviews generally focus more on assessing your thought process, and may even limit you to writing pseudo code.

Cultural fit: In addition to technical assessments, companies often prioritize interviews focusing on cultural fit. These conversations provide the prospective team with the chance to understand how your values align with the company culture. Expect questions that delve into your work style, collaboration preferences, and how you approach challenges as part of a team. Demonstrating your adaptability, communication skills, and enthusiasm for collaborative work is key to making a positive impression in these cultural fit interviews.



Preparing for an interview

Moving to the interview stage is a pivotal moment for your job search and can often come with nervousness or stress. Proper preparation is key to presenting yourself as a confident and capable candidate. This section will provide some essential steps to ensure you navigate the interview process seamlessly and leave a lasting positive impression on potential employers.

- Respond promptly:** When contacted by a hiring manager or recruiter for an interview, respond promptly. Don't feel pressured to respond outside of regular working hours, however, demonstrate your enthusiasm and commitment by acknowledging their outreach in a timely manner.


- Share your availability:** Many companies use special applications that allow you to self select your availability, but if this isn't the case, provide a range of dates and times for the interview within the upcoming weeks. If dealing with different time zones, specify your current time zone to avoid scheduling confusion.

- Time your availability strategically:** Whenever possible, schedule the interview on a date and at a time when you have few or no other commitments. This minimizes stress and allows flexibility for the interview to extend if needed.

- Present yourself professionally:** Regardless of the interview format (in person or online), present yourself professionally. While game industry dress codes may lean toward casual, research the company's expectations and opt for business casual attire if uncertain. This said, do not overdress for the interview. Rarely is a suit and tie expected in games, and can communicate a lack of research into the industry.

- Stay positive:** Avoid excessive negativity, even if your job search has been challenging. Present yourself as genuinely excited about the opportunity, focusing on a positive mindset; remember, this interview might lead to a job offer.





Online interview etiquette: If your interview is online, be sure implement the following guidelines:


- Choose a quiet location to avoid interruptions.
- Test your camera, microphone, and audio in advance to prevent technical issues.
- Keep your phone and computer plugged in, or have your device chargers nearby.
- Pay attention to the background, ensuring it is neat and presentable.
- Consider using a professional digital background if necessary.

Practice interview: If you feel nervous, consider conducting a practice interview. This helps familiarize yourself with common questions and boosts your confidence. This can be done with a trusted friend or family member, or simply by answering example interview questions out loud by yourself.

The STAR interview method

The STAR method, which stands for Situation, Task, Action, and Result, is a common approach where interviewers often frame questions to be best addressed using this structured format.

Watch for questions that prompt you to describe past situations, discuss specific challenges, or detail achieved results. When responding, structure your answers to articulate the situation or task, the actions you took, and the positive outcomes attained. This method provides a systematic way to highlight your problem-solving and decision-making skills, aligning seamlessly with the industry's interview expectations. Utilizing the STAR method enables you to stay focused, respond succinctly, and demonstrate your skills with the interviewer's preferred format, leaving a lasting positive impression.



Navigating job rejection

Rejection isn't personal: Job hunting is tough, especially when facing rejection or lack of responses. Remember, these setbacks don't define your self-worth or skills. They are often part of the process and not a reflection of your abilities or value.

It's a numbers game: With sometimes hundreds of applicants for each job opening, resumes can easily be overlooked. Rejections often stem from high competition and timing, not necessarily your qualifications.

Decision complexity: Employers often must choose from several strong candidates, meaning rejection doesn't always relate to your capability. It's often about finding the best fit among qualified contenders, so don't let this shake your confidence.

Persistence pays off: Job hunting requires consistency and perseverance. Rejection is part of the journey, but it doesn't determine your worth or future success. Use setbacks to refine your approach, learn, and continue applying confidently.

Seek feedback: Whenever possible, reach out for constructive feedback from recruiters to gain insights on how you interviewed, which will help you enhance future efforts. Remember, your aim is not just to land a job, but to find the right fit for both yourself and the employer.

Focus on growth: Use downtime between applications to improve skills, update your resume, and explore professional development opportunities. This shows potential employers your commitment to growth and boosts your confidence.



Acknowledgements

The development of this Universal Job Profile was made possible by the expertise and support of the Employer Advisory Board (EAB). Composed of professionals from leading companies in the real-time 3D landscape, the EAB serves as dedicated subject matter experts for the initiative, offering invaluable insights into the in-demand job roles within their respective industries. We extend our sincere thanks to each member of the EAB for their commitment to the success of the Universal Job Profiles. Their dedication not only showcases their professionalism but also highlights their significant investment in shaping a brighter future for the RT3D industry. We appreciate the collaborative spirit and contributions of the EAB, which have played a crucial role in advancing careers and opportunities within the real-time field.

Employer Advisory Board Members



With special thanks to:

With special thanks to: Alex Boyce, Anne Johnson, Brittany Gilbert-DeMarco, Dan Hewlett, Jason Harrison, Jason Parks, Julian Chelo, Lianna Johnstone, Lyle Maxon, Michael Courneya, Molly Kodros, Nick Janicki, Patrick Lenahan, Patrick Owens, Renee Gittins, Ricardo Arango, Ryan Cassidy, Sarvesh Navelkar, Stacey Long Genovese, Tanya Fraser, Turi Cacciatore, Ulises Pereida, William Garner, and Zak Whaley

About the Universal Job Profiles

The Universal Job Profiles are developed as part of **Elevate**, a Unity initiative dedicated to facilitating the entry of new talent into the games and creative 3D industries by establishing robust and open lines of communication among job seekers, educators, and employers.

Universal Job Profiles have been created to provide a unified framework for defining job roles within the games and creative sectors. The goal of this document is to serve as a handbook for anyone seeking a job, aiming to create a learning experience, or vetting candidates. By standardizing job roles, aspiring professionals can confidently acquire the necessary skills, educational institutions can design comprehensive learning experiences covering the full spectrum of each job, and employers can easily evaluate job candidates.

The data for Universal Job Profiles was gathered using the expertise of the Employer Advisory Board: a group of experts from industry-leading companies across all parts of the creative landscape, including games, media, training, and more. The board serves as our subject matter expert resource, providing crucial industry insights about in-demand job roles. By collaborating with the Employer Advisory Board, we ensure that the information shared in the Universal Job Profiles is up-to-date, accurate, and representative of actual industry needs.

These documents have been created in service to the games and wider creative 3D industries, aiming to enable more diverse and talented individuals to secure jobs in this dynamic field. As such, Universal Job Profiles will always be freely available for public use.

[To learn more, check out the Elevate page.](#)



Contributing to the Universal Job Profile

All Universal Job Profiles are living documents: they are reviewed by the EAB twice annually to ensure that they remain accurate and up to date with the latest needs of the games and creative 3D industries. We also welcome any suggestions from the community to help improve the overall quality and usability of these documents.

If you have any suggestions, questions, or feedback regarding this Universal Job Profile, please let us know by filling out this form:

[Universal Job Profile Feedback](#)

If you or your company has created a career development resource, such as a learning experience, certification or mentorship program that aligns with this Universal Job Profile and would like to have it included in this document, please fill out this form:

[Universal Job Profile course submission](#)

The Employer Advisory Board is actively recruiting new members. This is a volunteer board for companies that use game engines and other 3D tools to ship their products and personally employ staff that use these tool sets as part of their day-to-day job. Members of the EAB advise on industry standards, provide subject matter experts for informational interviews, and help determine what Universal job profiles should be made next. If your company is interested in learning more and potentially joining the board, please fill out this form.

[Employer Advisory Board Membership Application](#)



CHANGELOG



1.0-2025-02-18

- New pages added:
 - Key words
 - Internships
 - Job boards
- Updated skills format to better align to job listings
- Reorganized pages for better ease of use
- Update contact links
- Updated EAB membership logos

0.0.2 - 2024-06-25

- Early access release:
 - Minor layout adjustments
 - Updated contact links
 - Updated company logos
 - Added pay band info

0.0.1 - 2024-01-17

- Initial review release