AKYLADE®
Certified XR Developer™ (A/CXRD™) Exam Objectives

Exam Code: XRD-001

EXAM DETAILS

EXAM NAME	AKYLADE Certified XR Developer (A/CXRD)
EXAM CODE	XRD-001
QUESTION ITEMS	50 multiple-choice questions
DURATION	120 minutes
PASSING SCORE	700 points on a scale of 100 to 900 points

EXAM DOMAINS

	DOMAIN	EXAM COVERAGE
1.0	XR Developer Fundamentals	24%
2.0	Software Implementation & Integration	22%
3.0	Optimization & Performance Management	20%
4.0	Prototyping & User Experience (UX) Design	18%
5.0	Project Workflow & Best Practices	16%

IMPORTANT NOTE:

AKYLADE continuously reviews and updates exam content to ensure that exams remain current, relevant, and secure. When necessary, AKYLADE may publish updated exams based on the existing exam objectives, but all related exam preparation materials will remain valid during a given exam's lifecycle. Under each objective, a list of bulleted examples has been provided to aid candidates during their studies. These lists are not exhaustive and serve as examples of technologies, processes, regulations, or tasks related to the relevant objective. Other examples may appear on the exam, even if they are not listed directly under the objective in this document.

DOMAIN 1: XR Development Fundamentals (24%)

Candidates must be able to configure and navigate Unity or Unreal environments, script and manipulate 3-D scenes using sound physics and math, and package optimized builds for deployment on target XR devices.

1.1 Given a scenario, understand and utilize core XR development environments and interfaces.

- XR development environments
  - Unity
    - Scene view
    - Hierarchy
    - Inspector
  - Unreal Engine
- Multi-scene setups
  - Set up
  - Manage
- Unity's Package Manager
  - Add XR modules
  - Manage XR modules
  - Update XR modules
- Unity's Asset Store
  - Select pre-made assets
  - Import assets
  - Manage assets
- XR platform capabilities
  - Oculus
  - HTC Vive
  - HoloLens
- XR input systems
  - Head-mounted displays (HMDs)
  - Motion controllers
  - Hand tracking
- XR experience types
  - Augmented Reality (AR)
  - Virtual Reality (VR)
  - Mixed Reality (MR)

1.2 Given a scenario, apply programming principles to XR game development.

- C# gameplay scripting
  - Gameplay logic script
  - Trigger-based actions
- Object-oriented programming (OOP)
  - Classes and objects
  - Inheritance and polymorphism
  - Interfaces and encapsulation
- Event-driven programming
  - Events and delegates
  - Player actions
  - System triggers
- Game state management
  - Singletons
  - State machines
- Coroutines
  - Timed-based actions
    - Delays
    - Animations
  - Sequence control
- Custom Inspector scripts
  - Workflow enhancements
  - Inspector customization

1.3 Given a scenario, build and manipulate 3D environments in Unity.

- 3D scene composition
  - GameObjects
  - Lighting
    - Directional
    - Point
    - Ambient
  - Cameras
  - Object placement
- Player movement systems
  - XR character controllers
  - XR input integration
- Object transforms
  - Position
  - Rotation
  - Scale
  - Grid alignment
  - Snapping tools
- Rendering pipelines
  - URP
  - HDRP
- Render settings
  - Performance
  - Visual fidelity
- Asset management
  - Models
  - Textures
  - Audio assets
  - Format optimization
- Materials and shaders
  - Shader tools
  - Unity Canvas UI panels

1.4 Given a scenario, implement core physics and animation systems.

- Physics components
  - Rigidbody
  - Collider
- Physics responses
  - Force
  - Gravity
  - Collision
- Object pooling
  - Spawning
  - Destruction
- Raycasting
  - Object detection
  - Interaction triggers
- Animations and visual effects
  - Animator component
  - Animation component
  - Timeline sequences
  - Particle systems
- AI navigation and behaviors
  - Navigation Mesh (NavMesh)
  - Non-player character (NPC)

1.5 Given a scenario, apply mathematics and architecture in XR development.

- XR project setting configuration
  - Unity build settings for target platforms
  - Device-specific optimization
    - Oculus
    - Quest
    - HoloLens
- XR Camera and prefab architecture
  - Configure XR camera rigs
    - First-person
    - Third-person
  - Manage prefabs and reusable assets
- Asset usage optimization
  - Reduce memory usage
  - Improve texture/model performance
- 3D math fundamentals
  - Vectors
  - Matrices
  - Quaternions
- Spatial systems
  - Coordinate systems
    - Local
    - World space
  - Object transformations
- Data persistence
  - Implement serialization
  - Save and load structured data
- Spatial/3D audio systems

DOMAIN 2: Software Implementation & Integration (22%)

Candidates must be able to build intuitive interaction systems, integrate multi-platform rendering-audio-VFX pipelines, and implement gameplay, UI, and networking features that transform concepts into fully functional XR experiences.

2.1 Given a scenario, implement interaction systems for immersive XR experiences.

- Controller-based interactions
  - Map XR controller buttons
  - Grab and throw mechanics
  - Force feedback (haptics)
- Hands-free interaction systems
  - Gaze-based controls
  - Gesture-based interactions
  - Hand tracking
- Voice and audio-based interaction
  - Real-time voice modulation for character dialogue
- Voice commands and recognition
- Cross-platform input types
  - Touch
  - Gesture
  - Controllers
- XR interaction tools
  - Unity's XR Interaction Toolkit
  - XR Ray Interactor
    - Object targeting
    - Selection

2.2 Given a scenario, integrate XR systems for rendering, audio, and visual effects.

- Lighting and environmental realism
  - Light probes
  - Reflection probes
  - Object occlusion
  - Unity's Post-Processing Stack
- Visual effect integration
  - Particle systems
    - Sparks
    - Smoke
    - Environmental effects
  - Mixed reality capture
    - Real-world footage
    - Virtual elements
- Audio system integration
  - Spatial audio for 3D
  - Unity's Audio Mixer
- Cinematic and visual storytelling
  - Unity's Cinemachine
  - Unity's Recorder
- Immersion enhancement
  - Visual feedback
  - Audio feedback

2.3 Given a scenario, implement animations, physics, and object behaviors in XR.

- Object physics and behaviors
  - Collision-based events
  - Lifelike object movement
  - Physics-based interactions
    - Grabbing
    - Pushing
  - Hand-physics interactions
    - Grip mechanics
  - Object manipulation
- Animation systems
  - Character animations
  - Object animations
  - Timeline-based sequences
  - Animation transitions
  - Unity's Animator Controller
    - Blend animations
    - Trigger states
- Procedural animation
  - Animation rigging
  - Inverse kinematics (IK)
    - Realistic limb movement
- Hand presence and interaction
  - Animate finger movement
  - Gesture animation
  - Interaction feedback

2.4 Given a scenario, configure platform-specific and multi-device XR support.

- Spatial persistence
  - Create spatial anchors
  - Persist AR content across sessions
- Multi-platform deployment
  - Mobile
  - PC
  - Standalone XR
- Networked and multi-user experiences
  - Multiplayer XR setup
  - Real-time synchronization
- Cross-platform input systems
  - Unified input handling across devices
  - Device-specific adaptations
- Third-party integration
  - SDK/plugin integration
  - System-level service configuration
  - Hardware and device setup
  - Customize VR camera rigs
  - Device calibration
- Cloud-based services
  - Data storage
  - Leaderboard integration
- Platform frameworks
  - SteamVR
  - ARKit
  - ARCore
- Dynamic content loading
  - Asset bundles
  - On-demand asset delivery

2.5 Given a scenario, implement game mechanics, UI, and scene functionality.

- User interface design
  - Interactive UI elements
    - Buttons
    - Sliders
  - Custom shaders for XR visuals
- Gameplay mechanics
  - Teleportation systems
  - Locomotion and movement
  - Spatial mapping for AR
- AI systems
  - Companion AI behavior
  - Pathfinding with Unity's NavMesh
- AR-specific features
  - Image tracking
  - Marker-based object placement
- Voice interaction
  - Voice commands (hands-free)
  - Voice chat in multiplayer VR
- Scene and event scripting
  - Unity Timeline
    - Cutscenes
    - In-game events

DOMAIN 3: Optimization & Performance Management (20%)

Candidates must be able to profile and debug XR applications, optimize rendering, memory, and threading to maintain smooth frame-rates, and validate stability across diverse hardware and networked environments.

3.1 Given a scenario, debug XR systems for consistent functionality.

- Input system debugging
  - Controller bindings
  - Hand tracking accuracy
  - Input consistency across XR devices
- Multiplayer synchronization
  - Debug network latency
  - Identify sync issues in shared experiences
- Physics and interaction debugging
  - Object interaction behavior
  - Raycasting precision
  - Collision-related anomalies
- Rendering pipeline issues
  - URP to HDRP compatibility
  - Visual fidelity discrepancies
- Script and gameplay logic validation
  - Debug gameplay scripts
  - Identify logical bugs

3.2 Given a scenario, optimize rendering and graphics performance.

- Asset and texture handling
  - Optimize asset imports
  - Adjust texture compression
  - Balance detail and performance
- Lighting and shadows
  - Optimize settings
  - Baked lighting
  - Unity's Lightmapping
- Shader and material performance
  - Optimize material shaders
  - Improve real-time rendering
- Distance-based rendering strategies
  - Level of Detail (LOD)
  - Distant object performance
- Rendering optimization techniques
  - Adjust physics settings
  - Reduce draw calls
  - Implement batching
- AR/XR recognition and responsiveness
  - Optimize AR image tracking
  - Improve real-time responsiveness

3.3 Given a scenario, manage memory, threading, and asset performance.

- Memory management
  - Manage garbage collection in C#
  - Allocate memory for asset-intensive XR scenes
  - Use memory profiling
    - Detect memory leaks
    - Reduce memory leaks

Rendering performance
  - Manage occlusion culling
  - Optimize camera settings
    - Field of view
    - Depth of field
- Asynchronous loading and asset handling
  - Async loading
  - Unity's Addressables system
- Threading and computation
  - Multi-threading
  - Offload complex calculations
- Device performance monitoring
  - Temperature
  - Battery usage

3.4 Given a scenario, optimize animation, interaction, and audio systems.

- Visual and interaction enhancements
  - Optimize particle systems
  - Optimize UI elements
  - Resolve object clipping
  - Resolve occlusion issues
  - Physics issues in unique environments
    - Zero-gravity
    - Underwater
  - Audio performance
    - Profile audio playback
    - Efficiency
    - Responsiveness
- Animation optimization
  - Identify bottlenecks
  - Optimize playback
  - Optimize models

3.5 Given a scenario, test and profile across platforms and environments.

- Performance validation
  - Frame rate stability
  - Refresh rate consistency
  - Device and feature compatibility
  - Object interaction fidelity
    - Grabbing
    - Manipulation
- Cross-platform compatibility
  - Validate functionality
  - Test device-specific features
    - Controller types
    - Oculus Quest hand tracking
- Multiplayer performance
  - Network latency statistics
  - Accuracy of synchronization
- Environment and hardware testing
  - XR in various physical environments
    - Lighting
    - Space
    - Occlusion
  - Realistic object physics
  - Hand interactions
- Debugging and profiling tools
  - Frame-rate debugging tools
  - Unity Profiler

DOMAIN 4: Prototyping & User Experience (18%)

Candidates must be able to prototype and refine control schemes and interfaces, design spatially aware and accessible experiences, and conduct iterative user testing to ensure comfort, usability, and immersion.

4.1 Given a scenario, design and build user-centric XR interfaces and interactions.

- Onboarding and tutorials
  - Introduce XR controls
  - Explain features and mechanics
- Interface design
  - Create interactive menus
    - Dynamic elements
    - Immersive layouts
  - Outline user workflows with wireframes
- Visual feedback systems
  - Haptic response cues
  - Visual indicators for user guidance
    - Interactive objects
    - AR environments
  - Visual clarity and comfort
    - Color theory for accessibility
      - Scene readability

4.2 Given a scenario, prototype and evaluate XR control and input systems.

- Controller and input prototyping
  - Controller type variations
  - Touch interactions
  - Proximity-based interactions
  - Gaze-based navigation
  - Gesture-based controls
  - Voice control features
- Interaction prototyping
  - Gameplay mechanics
  - Facial interaction overlays
    - AR face filters
- Testing and iteration
  - Unity Play Mode
  - Refine based on feedback

4.3 Given a scenario, develop and test spatial awareness and comfort features.

- Virtual boundary systems
  - Safe zone design
  - Boundary collision testing
- Comfort and motion adaptation
  - Motion sickness reduction
  - Smooth locomotion options
- Spatial awareness tools
  - AR navigation aids
  - Environmental orientation features
- Camera and visual alignment
  - Adjust VR camera angles
  - Test visual comfort
  - Mixed reality transitions
  - Hand-off between VR and AR modes
- User immersion and presence
  - Avatar responsiveness
  - Balance events for user comfort
  - Test head movement clarity

4.4 Given a scenario, conduct user testing and usability evaluations.

- User testing and data collection
  - A/B testing for UI and UX designs
  - Field testing AR tracking in various environments
  - Conduct user feedback sessions
  - Track interactions with analytics
- Usability and accessibility evaluations
  - Heuristic evaluations
  - Accessibility testing
    - Visual
    - Auditory
  - Test diverse user input
    - Collaborative input
    - Diverse controller types
- Prototyping support and refinement
  - Use device simulators for hardware-agnostic testing
  - Use focus groups to evaluate preferences
  - Iterate character animations based on feedback

4.5 Given a scenario, design, document, and test accessible and inclusive experiences.

- Inclusive design strategies
  - Build interfaces for diverse user needs
  - Create inclusive tutorials
  - Design with accessibility in mind
    - Color contrast
    - Audio cues
- Testing for accessibility and usability
  - Screen-based testing for mobile XR
  - Validate functionality with test plans
  - Use collaborative input from diverse users
  - Evaluate accessibility features
    - Colorblind modes
    - Audio cues
- Iteration and refinement
  - Iterate designs with feedback
  - Refine UI for accessibility
  - Improve prototypes for usability
- Documentation practices
  - Document implemented features
  - Technical references for inclusive systems

4.6 Given a scenario, validate performance and feedback systems during prototyping.

- Evaluate locomotion and movement
  - Design locomotion methods
    - Teleportation
    - Joystick-based movement
  - Test comfort
  - Evaluate responsiveness
- Evaluate environmental realism and layering
  - MR environmental reflections
  - Test object occlusion layering
- Performance stability
  - Monitor frame rate consistency
  - Testing different gameplay scenarios
- Interaction reliability
  - Validate hand-tracking
  - Validate controller input
  - Assess latency
  - Access responsiveness
- Feedback systems
  - Test particle effects

Evaluate spatial audio positioning

DOMAIN 5: Project Workflow & Best Practices (16%)

Candidates must be able to manage version control and agile task workflows, coordinate cross-disciplinary collaboration with clear documentation, and enforce clean, scalable coding and production standards throughout the XR project lifecycle.

5.1 Given a scenario, apply version control and source management practices.

- Version control systems
  - Git source code management
    - Create branches
    - Manage branches
    - Merge branches
    - Rebase a branch
  - Platforms
    - GitHub
    - Bitbucket
- Branching and development workflows
  - Implement branching strategies
    - Feature development
    - Bug fixes
  - Maintain a clean commit history
- Code documentation
  - Track code changes
  - Document project architecture
  - Record technical decisions
- Collaboration and review
  - Code reviews
  - Pair programming
  - Feedback
    - Provide
    - Receive

5.2 Given a scenario, manage tasks, documentation, and Agile workflows.

- Agile project management
  - Agile methodologies
  - Scrum
    - Sprint
    - Sprint planning
    - Daily scrum (Stand-up)
    - Sprint review
    - Sprint retrospective
- Team collaboration
  - Track progress with task boards
  - Coordinate across teams for aligned execution
- Task tracking tools
  - Create and manage Jira tickets
  - Workflow tracking
    - Trello
    - Asana
  - Document user stories and milestones

5.3 Given a scenario, collaborate across disciplines to build XR experiences.

- Cross-functional collaboration
  - Holistic XR team contributions
  - Align design and development
- Audio design integration
  - Sync sound design with interactions
  - Communicate with audio specialists
- Level and environment design
  - Collaborate with level designers
  - Test and refine XR environments
- Backend engineering coordination
  - Cloud service implementation
  - Real-time features and data handling
- Art and asset collaboration
  - Work with artists
    - Models
    - Textures
    - UI
  - Ensure seamless asset integration
- Hardware and peripheral testing
  - Collaborate with hardware engineers
  - Support XR input and feedback devices
- Quality assurance (QA)
  - Partner with QA testers to identify bugs
  - Validate user experience across builds

5.4 Given a scenario, implement and utilize industry standards for clean, scalable development.

- Code quality and structure
  - Maintainable coding standards
  - Organize project hierarchy
    - Asset naming
    - Folder structure
    - Comment conventions
- UI/UX consistency
  - Apply UI/UX guidelines
  - Interfaces
    - Intuitive
    - Accessible
- Documentation
  - Project naming standards
    - Scripts
    - Assets
    - Prefabs
  - Align with existing documentation
- Design collaboration tools
  - Figma
  - Adobe XD
  - Sketch
  - InVision

5.5 Given a scenario, communicate effectively and conduct ongoing improvement.

- Team collaboration
  - Align technical implementation with design goals
  - Share progress and blockers with team
- Stakeholder engagement
  - Gather requirements
  - Integrate feedback into design
- Participate in creative collaboration
  - Brainstorming
  - Ideation
  - Feature development discussions
- Professional development
  - Stay current with tools and techniques
  - Explore emerging development trends
  - Compatibility testing
  - Validate with different versions
    - SDKs
    - Engines