Assessment and grading – Case studies

Paint a picture of how you do assessment and grading in your Unity classroom. How do you create an environment where students can make mistakes, while still maintaining a high level of rigor?

Have a resubmission policy
I created a rubric that explained expectations for deliveries. That rubric was fairly consistent for all hand-ins, but for every hand-in, I gave the opportunity to correct and deliver any feedback to improve scores, as long as it was two days before my grades deadlines.

Align assessment to industry standards
For our Unity courses, typically we look at the following elements:

- Game mechanics - did it function and did it have the required elements?
- Gameplay dynamics - did the game have a good challenge flow, were the choices meaningful, were the consequences appropriate, was a feedback loop implemented?
- Player engagement - was there a narrative (did it make sense)? Were the visuals and audio cohesive to the game atmosphere?
- Playtesting - students play each game and provide their own feedback, the playtest scores are then averaged and tallied in the overall rubric.
- Publishing - was the game submitted in the correct format with the correct details?

“Failure is an option”

In most classes, students work on the projects in sprints, and they should be able to fix any bugs before the final submission (submitting a functional game level). In our exploration class, I start by stating that "failure is an option". The course is always about exploring new tools and techniques. They have four weeks to create a prototype and then present what they have. In this presentation, they may discover that the tool/feature they were implementing wasn't quite ready for production or more resources would be needed to fully implement it as intended.

Gamify assessment using sprints
Course evaluation uses a gamified approach: completing eight sprints of the project earns the student a grade of eight (B). The student can do extra sprints after the course finishes, for up to one month. We apply an additional rubric for the criteria (gameplay, aesthetics, and programming). Each category can raise the final grade to 10 (A+). The student can decide to do only six sprints during the course. They will get the opportunity to finish their project next year.

Mix formative and summative assessments
I blend formative and summative assessments. Skills-lessons and prototypes are formative, challenges/quizzes and personal projects are summative.

Use the principles of continuous assessment (CA)
My Unity modules are 100% continuously assessed (CA) with no final exams. Normal CA breakdown is as follows:

- 20%: Practical evaluation - E.g. Complete Unity learning pathways and challenges.
- 30%: In-class practical Exam - Build and publish a demo scene to a specific specification in a two hour exam.
- 50%: Individual project - Research, design, build, and test a VR experience.

One of the ways in which students can make mistakes is by creating challenges within the same exercises proposed in the course. To review exercises, I do use the rubrics within the course.

Reward effort and continued participation with extra perks
Reward students who share their projects and participate in events. I do this by giving them assets, software keys, or notes on developments I am involved in that might help them.

Have a bug jar
Have a candy or other prize jar students can pull from when they solve a problem or find a way to pivot after failure. This helps incentivize effort and productive struggle.

Use a grit-based rubric
Encourage perseverance through a grit-based rubric such as this one from Twitter.