Educational Software

Katie Vale, Ed.D.
Director of Academic Technology, HUIT
CS50, 11/6/11

When is educational software particularly useful?

- Motivation for different learning styles
- Simulation and role play (including games)
- Visualization (scientific inquiry, 3D modeling)
- Collaboration (wikis, shared documents)
- Interactive drill and practice (direct instruction)
- Critical thinking (data analysis tools)
- Learning by doing (students build software or movies)

When is it less useful

- “I wanted an excuse to learn Ruby/make a mobile app/etc.”
- When it isn’t integrated into a course
- When it’s an optional assignment
- When it’s more style than substance (“iPads are cool!”)

How you might think it works

- Take a CS class
- Have an idea
- Write code
- Give app to a teacher
- ???
- Profit!

- Sadly, no.

How it really works

- Identify an educational problem or set of goals with the instructor
- Identify criteria for success (i.e. tests)
- Create the materials and choose pedagogical methods and media (lectures, assignments, etc.)
- Pilot the software and tweak as necessary
- Document and port code in future as needed

How to write specs

- “After using this software, students should be able to…”
- Use action verbs in completing this sentence: “demonstrate, list, compare, discern, identify” instead of passive verbs “understand, appreciate, learn”
- This can be tricky – another approach is to ask “what should students remember from this app in a few years?”

Examples

- "After using these materials, students will be able to":
  - identify parts of a schematic circuit diagram
  - categorize musical works by American jazz musicians
  - list the phonologic differences between Quebecois French and Caribbean French

- Storyboard the action if it's a game or activity

Non-ideal Examples

- "After using these materials, students will be able to":
  - understand statistics better
  - have an appreciation for Vogon poetry
  - know how wireless networks work

- These are all valid aims, but they are hard to assess – ask "how will you know if this has happened?"
- You may have to push the teacher to arrive at measurable goals.

Think about usage

- How will the software be used? By self-paced learners? In class? Homework? As group exercise?
- How will it be graded, if it will?
- If the teacher thinks of it as an ungraded optional exercise, run!
- Custom educational software is time-consuming and expensive.

Other considerations

- What do the students have for technology? Design for the lowest common denominator, even if that's no fun.
- What training will the students need? Who will provide it? Will the instructor need help too?
- Can you use control groups? Grants may require them, plus assessment data.

Most important success factors

- Engaged teacher who sees value of project
- Explicit educational goals
- Commitment over years (funding, porting)
- Good documentation (both user and code)
- Design that can outlast you – don't try to make yourself indispensible
- Note: the average lifetime of custom educational software is less than four years

Development platforms

- Will change constantly – design assuming you will one day have to port it…several times
- My first project: Guide to Intermedia to Hypercard to museum installation within 4 years
- The best advice is use civilized coding practices – document your code!
- Accept that teaching needs change and eventually your project may be retired

Has it been done already?

- Check repositories like http://www.merlot.org, http://www.nmc.org, app stores and http://atgportfolio.fas.harvard.edu
- Look through these before developing, you might save yourself some time.

Do you have to write something?

- Is writing software the best way to achieve the goals? What about video (e.g. Khan Academy or Instructables), a custom text, or a hands-on project?
- Often the best choice is to repurpose existing software within a lesson (e.g. Google Maps, Matlab, Piazza, Excel, a wiki)
- Don't lead with the technology (e.g. "Can Farmville be used for teaching?")

Careers in educational technology

- Applications developer
- Educational publishing
- K-12, library, museum teaching or media specialist
- University teaching or academic technology
- Corporate and military e-learning and training
- Broadcast and interactive media

Want to learn more?

- Student employment opportunities
- UTEP and TIE programs at GSE
- Contact me if you'd like to discuss ideas, graduate programs, etc.:
- katie_vale@harvard.edu

- Thanks and enjoy CS50!