SpaceChem

Michael James Scott, Games Academy, Falmouth University, United Kingdom
michael.scott@falmouth.ac.uk

Game: *SpaceChem*
Developer: Zachtronics
Year: 2017
Platform(s): Windows; Mac OS; Linux; iOS; Android
Number of players: Single player
Genre: Logic; puzzle
Type of game: Computer-based digital game
Curricular connections: Computer science
Possible skills taught: Computational thinking; problem solving; algorithm design
Audience: 14+ (high school students, college students)
Length of time: 60 minutes per session; 20+ hours outside of class
Where to play: Computer lab; room with tablets/laptops; home
Cost: Educational licenses available for free
URL: http://www.zachtronics.com/spacechem/
Tags: computer science; programming; computational thinking; coding; problem solving

Summary

SpaceChem is a 2D logic puzzle game designed by Zach Barth. Players assume the role of a spacefaring reactor engineer who is responsible for setting up new reactors across a series of planets. This involves challenging players to design processes that take sets of chemical compounds as inputs and converting them into specified forms as outputs. Although the game is themed around futuristic chemistry, its simplicity serves abstract and symbolic purposes. The core game mechanic centers upon algorithm design. As illustrated in Figure 1, players program the reactor using a visual drag-and-drop interface. Players drag instructions from the menu at the bottom of the screen, and place them onto the reactor grid, which is reminiscent of a top-down view of a claw crane arcade machine. The system deposits chemicals into the input areas (marked alpha and beta). So, when players start the reactor by selecting the play icon, the "waldos" (the claws that grab and drop the atoms, marked in red and blue) move around the reactor according to the instructions that they have placed in each cell. For example, the turn instructions (marked W, A, S, and D) allow players to define the path of each waldo. The grab/drop instruction (labelled E) makes the waldo pick up or deposit the chemical on that particular cell. As the game progresses, it introduces other reactor components that perform operations such as: bonding; sensing; teleporting; fusing; and fissioning. These activate when a waldo moves over a cell containing the corresponding instruction. Once all of the operations execute to produce the desired chemical, players must ensure that their program deposits the chemical into the designated output area (marked psi and omega).

[image_14.1]
Figure 1. Example circuit for a custom level in *SpaceChem*

How to Use the Game

It is often unclear how educators can effectively nurture computational thinking skills early in a computing course, particularly for those students who are new to computer science and have not yet mastered a programming language. *SpaceChem* presents an opportunity for students to develop these foundational skills in a problem-oriented context, without the burden of setting up and learning a sophisticated software development kit or the need to make use of a fully featured programming language.

Students can play through the different scenarios that the game presents both in class and at home. For instance, educators could lead students through the tutorial levels step-by-step, and as the game opens up, educators can then assume roles as facilitators, rendering aid to those students who struggle. After which, the students could then complete the remaining levels as a homework assignment.

After an initial walk-through, a valuable starting activity is circuit interpretation using the 'peer instruction' technique (see Simon *et al*, 2010). Show students a circuit and task them with tracing it to predict what will happen. Clickers systems that present multiple-choice questions can be useful for this. Have students vote individually, without discussion in the first instance. Reveal the breakdown of responses, and let students discuss for a few minutes and then revise their prediction. Then discuss the incorrect answers (i.e., the distractors) and possible misconceptions before showing the execution of the circuit.

As students start to play the game themselves, the learning environment will typically become quiet with few distractions. This will help many students engage with the problems that the game presents. However, many novice programmers will experience frustration. Particularly, if they have not done any programming before and so do not fully understand why their circuit designs fail. To alleviate this, educators can foster collaboration by having pairs of students share a computer and engage in pair programming (see Zarb, Hughes & Richards, 2013). Matching students of similar ability, ensuring that the 'driver' (at the keyboard) and the 'navigator' (away from keyboard) swap roles at regular intervals, and sharing guidelines will maximize engagement. Permit quiet discussion among pairs or small groups of students.

When students are stuck, facilitate their problem-solving approach away from the keyboard. Use a flipchart or mobile whiteboard to illustrate concepts and work through solutions. Avoid touching their mouse or keyboard unless necessary. A key benefit of *SpaceChem* is its ability to conveniently record and display solutions. This enables educators to compare the differing solutions that their students produce. Solutions for the same level are often quite diverse and perform differently against in-game metrics such as the number of steps, number of instructions, and number of reactors needed. This can help emphasize the notion that there is no single correct answer. Additionally, students can often improve the solutions they initially propose, and can realize that different solutions might be suited for different needs, or measures of quality.

Researchers recognize the potential of games for computing education (Esper, Foster & Griswold, 2013) and continue to explore how they integrate into the classroom (see Johnson et al, 2016). To this end, academics at Falmouth University have been using *SpaceChem* as the first exercise in their undergraduate "Principles of Computing" class for several years. To help ensure participation, academics assess the task as coursework on a pass-fail basis. A pilot study in 2015 showed an improvement in computational problem-solving skills after playing the game across the first three weeks of the class (measured using a pre-post matched in-session problem-solving exercise; for further details, see Scott, 2017). The effect size was greater than one standard deviation from the mean. This finding suggests that *SpaceChem* can be effective in introductory computing classes.

Tips & Best Practices

1. Instead of a live walkthrough, teachers could consider providing students with a video walk-through that they can watch at their own pace.
2. The "record solution" feature that can be useful for additional show and tell and peer review type activities, where students to present their work to each other and provide each other formative feedback. Note that the default save location is the desktop.
3. *SpaceChem* has a sandbox mode called *ResearchNet* that can allow educators to create custom levels and will permit advanced students the ability to define more advanced challenges for each other.
4. Unlike university-level classes that might challenge students to complete the whole game, high school-level classes might focus on the tutorials on the first few planets before moving onto problems defined via *ResearchNet* to delay or avoid the more advanced operators.
5. The save files are SQLite databases, which record all in-game actions. Technically-minded educators can parse these records to investigate the problem-solving approach of each of their students to diagnose the challenges they encounter (and, if appropriate, identify plagiarism).

Related Games & Media

Infinifactory (http://www.zachtronics.com/infinifactory/)
Opus Magnum (http://www.zachtronics.com/opus-magnum/)
TIS-100 (http://www.zachtronics.com/tis-100/)
SHENZHEN I/O (http://www.zachtronics.com/shenzhen-io/)
Human Resource Machine (http://tomorrowcorporation.com/humanresourcemachine)
7 Billion Humans (http://tomorrowcorporation.com/7billionhumans)
[the Sequence] (http://ombgames.com)
Marvellous Inc. (https://marvellous.itch.io/)
Factorio (https://www.factorio.com/)

Further Reading

Barth, Z. (2011) "SpaceChem: A Guide for Educators" [Online] Available from: http://www.zachtronics.com/images/SpaceChem%20-%20A%20Guide%20for%20Educators.pdf

Esper, S., Foster, S.R., Griswold, W.G. (2013) "CodeSpells: Embodying the Metaphor of Wizardry for Programming," Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education.

Johnson, C., McGill, M., Bouchard, D., Bradshaw, M.K., Bucheli, V.A., Merkle, L.D., Scott, M.J., Sweedyk, Z., Angel, J., Xiao, Z. & Zhang, M. (2016) "Game Development for Computer Science Education," Proceedings of the 2016 ITiCSE Working Group Reports.

Papastergiou, M. "Digital game-based learning in high school computer science education: Impact on educational effectiveness and student motivation." Computers & Education 52.1 (2009): 1-12.

Simon, B., Kohanfars, M., Lee, J., Tamayo, K. & Cutts, Q. (2010) "Experience Report: Peer Instruction in Introductory Computing". Proceedings of the 41st ACM Technical Symposium on Computer Science Education.

Scott, M.J. (2017) "Games and Game Jams: An Employability-First Approach to Educating Programmers," Proceedings of the HEA Conference on Learning and Teaching in STEM Disciplines.

Zarb, M, Hughes, J. & Richards, J. (2013) "Industry-inspired Guidelines Improve Students' Pair Programming Communication," Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education.