Linting and Hints in Candli

Bringing Linting and Adaptive Program Completion to Candli, a Visual Game Programming Environment for Children

Stéphane Magnenat[1]

[1] Enlightware GmbH, Zurich, Switzerland

Abstract

Candli is a web-based educational game creation app in which children acquire STEAM skills by creating video games based on their own drawings. Candli allows children to program games using an accessible visual programming language inspired by established research traditions such as Scratch and Thymio VPL. While such languages are highly expressive, enabling rapid creation of dynamic and engaging agent behaviors, this very efficiency can pose challenges in classroom settings. In particular, children often encounter non-trivial problems quickly, leading to teacher overload and hindering learning. To mitigate this issue, Candli integrates technical scaffolding features designed to support children's autonomous learning. In this workshop, we explore two such features: linting and adaptive program completion hints. The linting feature draws inspiration from software engineering practices, providing immediate feedback to students about potential errors in their code – a feature rarely utilized in educational contexts. Adaptive program completion hints employ classical AI reasoning to incrementally guide students based on their existing code structure. This method involves comparing the current student code with predefined Prolog-like solution rules and suggesting minimal, targeted corrections that do not compromise other program functionalities. These features assist learners in developing accurate notional machines for understanding code execution while preserving opportunities for free exploration and playful tinkering. Collectively, these tools offer a concrete realization of constructionist learning principles.

Keywords and Phrases: Adaptive Learning, Visual Programming, Game-Based Education, Constructionist Learning

Workshop Description: Scope, Motivation, and Background

Candli (cand.li) is a web-based educational game creation app in which children acquire STEAM skills by designing and programming video games based on their own drawings and sound recordings. Candli represents the transfer of academic research (Chatain et al., 2019) into practice through our spin-off company Enlightware. It allows children to program using a visual language inspired by well-established platforms such as Scratch (Resnick et al., 2009) and Thymio VPL (Shin et al., 2014).

Magnenat, S. (2025). Linting and Hints in Candli. Bringing Linting and Adaptive Program Completion to Candli, a Visual Game Programming Environment for Children. *Constructionism Conference Proceedings, 8/2025*, 577–579. https://doi.org/10.21240/constr/2025/72.X

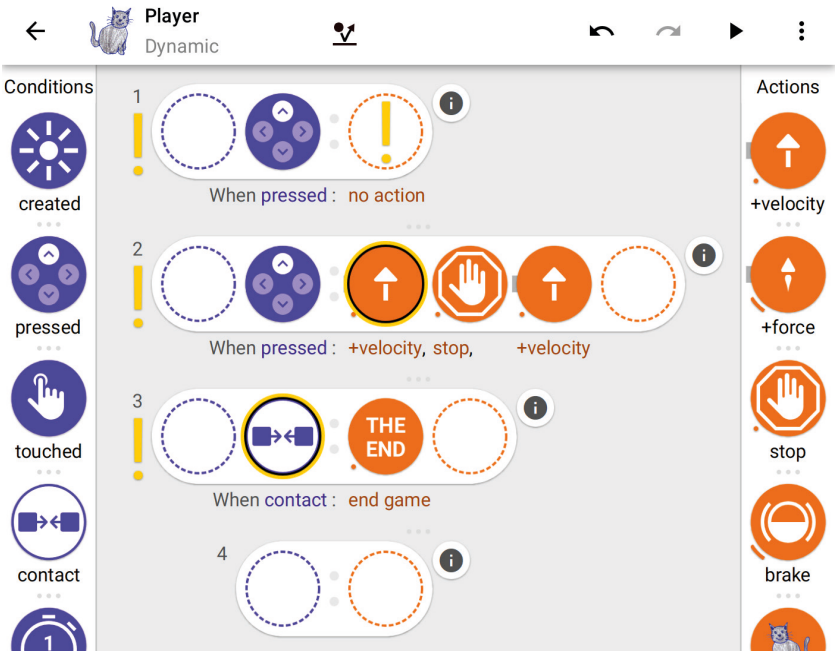

Figure 1:

Through classroom testing, we observed that while these visual languages are highly expressive – enabling rapid creation of dynamic and engaging agent behaviors – this very efficiency can pose challenges in classrooms with 20 or more students. In particular, children often encounter non-trivial problems quickly, which can lead to teacher overload and hinder the development of a coherent notional machine (Sorva, 2013) for understanding game behavior and execution.

To address these issues, we developed two key support features:

- Linting for visual programming languages. Inspired by practices in software engineering, Candli's linting system provides immediate feedback on potential code issues – an approach still rare in educational environments. For example, the system warns students if an action is duplicated needlessly (e.g., two self-destruction commands in a row) or canceled out (e.g., applying an impulse followed immediately by a stop).
- Adaptive program completion hints. These use classical AI reasoning to offer incremental guidance based on the student's current program. The system compares the student's code to a set of Prolog-like rules representing valid solutions, and then proposes minimal, non-disruptive changes that preserve existing functionality. Suggestions are conveyed through an interactive "Candli hand" that visually demonstrates the required user-interface actions to complete the code.

Based on qualitative classroom feedback, we believe these aids support learners in developing a more accurate notional machine while still allowing for playful experimentation. As such, they embody key constructionist learning principles.

These features are not without limitations. For example, in agent-centric, block-based environments like Candli, implementing even a simple behavior often involves coordinating multiple objects. While the adaptive hints can switch between objects, this may obscure the conceptual unity of the algorithm being taught. We see this as a limitation of the object-focused programming paradigm itself, but recognize the

need for better user-interface and user-experience design to help learners retain a sense of algorithmic coherence across multiple agents. This remains an area of active research for us.

In this workshop, we will present these features through hands-on play with Candli, just as children experience it. We aim to foster discussion around their benefits, limitations, and the broader design space – hoping to spark new research questions and design innovations within the constructionist learning community.

References

Chatain, J., Bitter, O., Fayolle, V., Sumner, R., & Magnenat, S. (2019). A Creative Game Design and Programming App. In Proceedings of the 12th ACM SIGGRAPH Conference on Motion, Interaction and Games. ACM.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Sonenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). Scratch: programming for all. *Communications of the ACM*, *52*(11), 60-67.

Shin, J., Siegwart, R., & Magnenat, S. (2014). Visual programming language for Thymio II robot. In *Conference on interaction design and children* (IDC'14). ETH Zürich.

Juha, S. (2013). Notional machines and introductory programming education. *ACM Trans. Comput. Educ*, 13(2), 1-31.