Mastery Grids for Python

Personalized Adaptive Web Systems Lab, School of Information Sciences, University of Pittsburgh
Learning+Technology research group, Aalto University, 2015

Mastery Grids is a system that allows students and teachers to view(monitor) their learning progress and knowledge in comparison with pairs or groups, create and access multiple smart learning contents according to students' levels, design a course curriculum with desired topics and matching learning materials. Our past research has successfully deployed Mastery Grids in various learning domains (Java, Python, SQL) in different universities. Python programming language is one of the already-covered learning domains in Mastery Grids. The Python programming course in Mastery Grids has been used for undergraduate and graduate level course since 2013. Although the course material covers an extensive range of Python programming topics, it can be extended using the authoring tools that come along with Mastery Grids. Our research shows that Mastery Grids effectively increases learners' performance, motivation, engagement and retention.

❑ Mastery Grids Interface Features

- ❑ **Knowledge and Progress Visualization:** Presenting student learning progress and knowledge by colored grids from four perspectives as shown in Figure 1: students' perspectives ("Me" or "Group") and comparison perspectives.
- ❑ **Social Comparison:** Implemented by "Me vs Group" and "Me vs other learners" in the visualization in various granularities including by topics or by learning content type (Figure 1, Figure 2).
- ❑ **Adaptive Navigation Support:** Recommendation of learning resources are provided based on student activities and learner knowledge in a centralized user modeling server. (see red stars in Figure 1)

Figure 1: Mastery Grids' knowledge and progress visualization of Me, Me vs Group, Group, other learners embedding open learner model of oneself and social comparisons of a student with the group or other learners.

Figure 2: Mastery Grids' social comparison by learning content (resource) type.

❑ Python Learning Smart Content and Flexible Course Design

Mastery Grids provides multiple types of interactive, smart learning contents including parameterized Python quizzes, annotated examples, and animated examples. Learning material in Python course are organized in topics. Topics are aligned in accordance with learning timeline as is represented in Figure 1.

- ❑ **Parameterized Semantics Problems:** Short-answer quizzes that test for students' understanding of codes. These questions can be repeated by the students, every time with different parameters. (Figure 3 Left)
- ❑ **Parson Problems:** Problems that ask students to drag different fragments of a code to construct a complete code in order to achieve a task. (Figure 3 Right)
- ❑ **Annotated Examples:** Short snippets of code with explanation of key lines in them. (Figure 4 Left)
- ❑ **Animated Examples:** Code examples that include graphics representing the variables and output screen. The student can run line by line and observe outputs in an animated fashion. (Figure 4 Right)

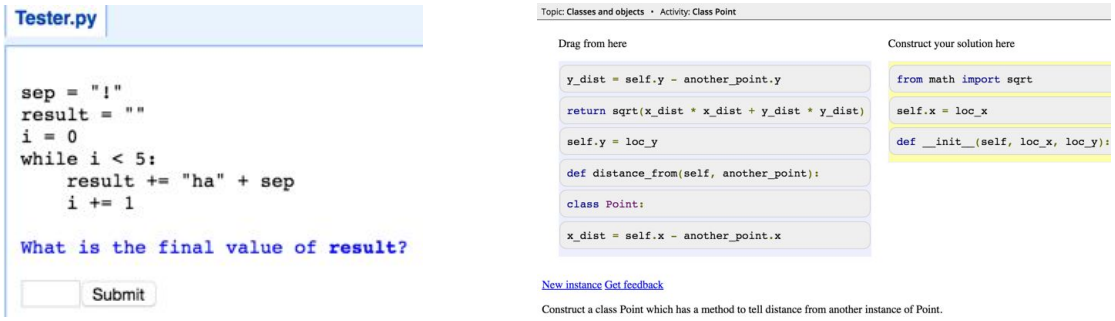

Figure 3: The left side shows a parameterized semantics problem, and the right side shows a Parson problem.

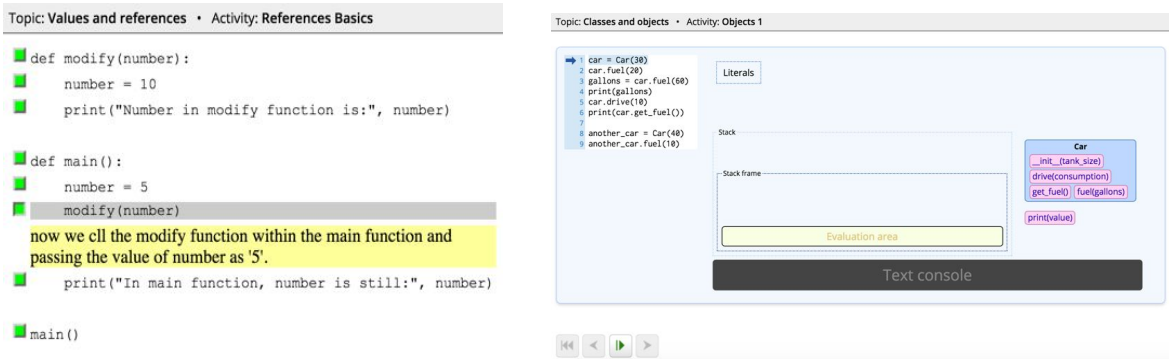

Figure 4: The left side shows an annotated example, and the right side shows an animated example.

- ❑ **Authoring Tools:** We have developed a set of authoring tools enabling the above content creation and a complete course design using these content. Tools include Content Authoring, Course Authoring, Group Authoring and 4) Authoring Portal to access different authoring tools.

❑ Learning Analytics Enabled

Mastery Grids with its architecture provides following major advantages for conducting learning analytics: 1) it allows flexibly experimenting different learner models, recommendation/personalization techniques, 2) its fine-grained logging mechanism for user activities allows for behavior analysis such as sequential pattern mining. 3) it allows easily incorporating new smart content on which one can conduct experiments to study the effect.

Our project website with detailed information can be accessed here bit.ly/OSLM_MG. All related code can be downloaded from Github: https://github.com/PAWSLabUniversityOfPittsburgh, https://github.com/acos-server/.