Integrating
Drawing Tablet and Video Capturing/Sharing to Facilitate Student Learning

ACM Global Computing Education
CompEd'19 / May 18 / Chengdu, China

Chen-Wei Wang
York University, Toronto, Canada

Challenges of Undergraduate Teaching

1. ***complex computational thinking***: ***limited prior exposure***
   - e.g., nested loops on 2D arrays [paper]
   - e.g., OOP: ***aliasing***, polymorphism, dynamic binding [talk]
2. ***scheduled in-class lectures***: ***limited comprehension***
   - Large class size restricts **pauses** and **interactions**.
   - Instructor's verbal **remarks** and written **notes** reflect their *insights into the taught subjects*, but . . .
     it's difficult to **copy** and **understand** them simultaneously.

How Would You Help this Upset Student?

Student:
I ***did attend*** classes
but ***failed to follow completely***.

Motivating Question

How can we make the

in-depth ***illustrations*** in class *accessible* to students

for their ***self-paced study*** outside the classroom?

Contribution: An Approach for Effective After-Class Learning

A technique for

- ***In-class illustrations*** of ***complex ideas*** on a *drawing tablet*.
  - **Pre-class** preparation of *starter artifacts* (e.g., code fragments)
  - **Frequent** and **heavyweight** *annotations*
- Allowing students to *review* taught contents outside class

Let's illustrate the technique using a short review lecture on OOP.

At the end of the lecture, ask me a question (**as a student**)!

Example Lecture: Class Model

Consider the following *model* of a person:

```
public class Person {
  /* Attributes */
  double weight; /* kilograms */
  double height; /* meters */
  /* Constructor */
  Person (double weight, double height) {
    this.weight = weight;
    this.height = height;
  }
  /* Accessor/Getter: Body Mass Index */
  double getBMI() {
    double bmi = this.weight / (this.height * this.height);
    return bmi;
  }
  /* Mutator/Setter: Change of Weight */
  void gainWeight(double amount) {
    this.weight = this.weight + amount;
  }
}
```


Example Lecture: Console Tester

What are the *console outputs* produced by the following test?

```
1  public class PersonTester {
2    public static void main(String[] args) {
3      Person jim = new Person(72, 1.72);
4      Person jonathan = new Person(65, 1.81);
5      System.out.print("Jim's BMI: ");
6      System.out.printf("%.2f\n", jim.getBMI());
7      System.out.print("Jonathan's BMI: ");
8      System.out.printf("%.2f\n", jonathan.getBMI());
9      jim = jonathan;
10     jim.gainWeight(3);
11     System.out.println("===== After Jim gained 3 kgs =====");
12     System.out.print("Jim's BMI: ");
13     System.out.printf("%.2f\n", jim.getBMI());
14     System.out.print("Jonathan's BMI: ");
15     System.out.printf("%.2f\n", jonathan.getBMI());
16   }
17 }
```


Example Lecture: Console Output

- Let's first verify this in Eclipse!

```
Jim's BMI: 24.34
Jonathan's BMI: 19.84
===== After Jim gained 3 kgs =====
Jim's BMI: 20.76
Jonathan's BMI: 20.76
```

- After Jim gained weight:
  Q: Why was Jim's BMI ***decreased***?
  [Didn't Jim ***gain*** weight?]
  Q: Why was ***Jonathan***'s BMI increased?
  [Wasn't it ***Jim*** who gained weight?]
- Let's illustrate how this happened!

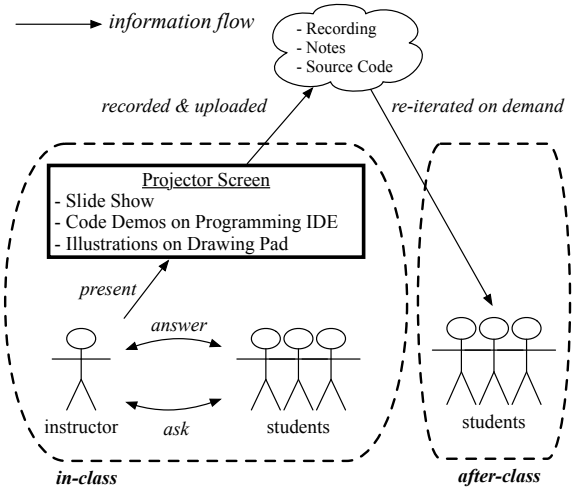
Example Lecture: Q & A

Questions about the OOP lecture?

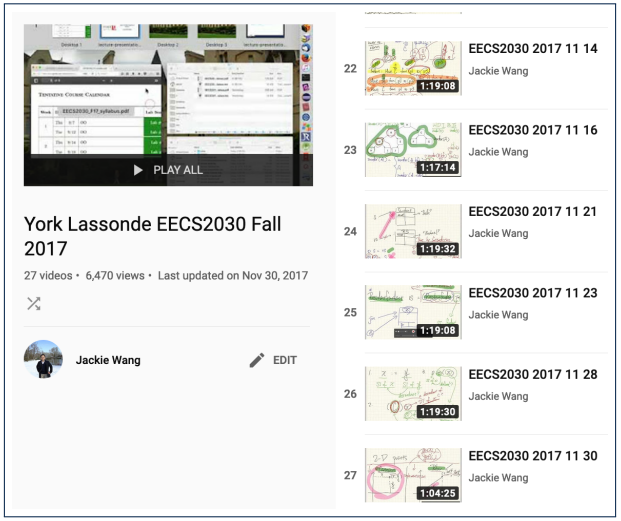
A Pattern for Teaching Complex Ideas

- I just demonstrated a *teaching pattern*, choreographing:
  - **Slide Show**: ***Specify*** Problem.
  - **Programming IDE**: ***Illustrate*** Solution.
  - **Drawing Tablet**: ***Annotate*** on starter pages to ***gradually*** build towards the solutions or conclusions.
    e.g., **starter** page vs. ***annotated*** page in the example lecture
  - **Drawing Tablet**: ***Answer*** students' questions.
- More examples:
  - Paper: teaching computations on 2-dimensional arrays
  - My lectures page: `https://www.eecs.yorku.ca/~jackie/teaching/lectures/index.html`

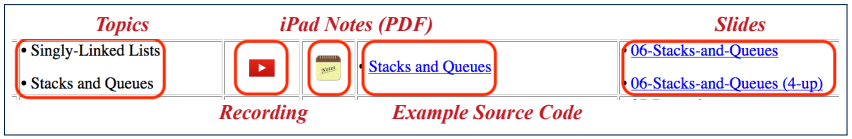
Contribution: An Approach for Effective After-Class Learning



Study Resources for Students (1)



Study Resources for Students (2)



Teaching Context

Proposed approach adopted in *undergraduate teaching*:

7 iterations of four courses [1st-, 2nd-, 3rd-year]

Taught ***1,295 students***

- ***Procedural Programming***
  - variables, assignments [**data flow**]
  - if-statements, loops [**control flow**]
  - arrays, linked lists, trees [**data structure**]
- ***Object-Oriented Programming***
  - classes, attributes, methods, objects, aliasing [**basic OOP**]
  - inheritance, polymorphism, dynamic binding [**advanced OOP**]
- ***Software Design***
  - design by contract, program correctness [**specification**]
  - design patterns [**architecture**]
- Nonetheless, the proposed approach is *sufficiently general* for teaching any ***complex idea***.

Reflections

- Instructor's Efforts

  Starter Pages: What concepts/examples should be illustrated?

- Drawing Tablet vs. **Blackboard/Whiteboard**
  - ***Time Effectiveness***: Pre-set starter pages save time on copying.
  - ***Reusability***: Starter pages may be elaborated and reused.

- Drawing Tablet vs. **Slide Animations**

  Flexibility: ***Dynamic*** control of the pace and level of details w.r.t. the ***comprehension level***.
  e.g., *starter* page vs. *annotated* page in the example lecture

- Review of Lectures

  Repetition: Even effective in-class illustrations take repetitions to achieve ***full comprehension***.

Beyond this talk. . .

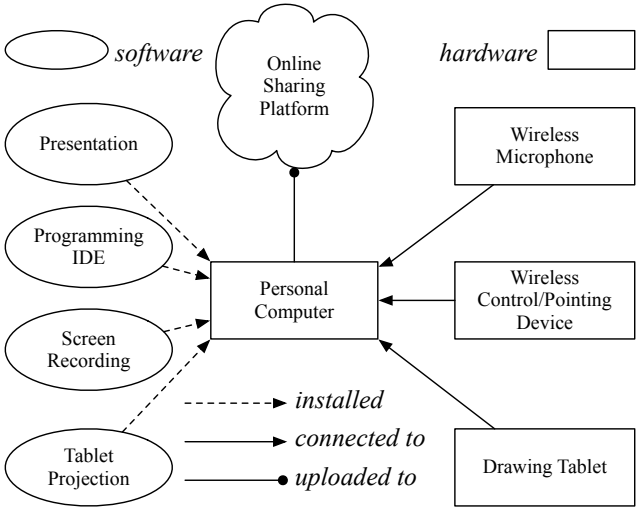
- Read my paper!
  - Adopting the Approach
  - Evaluation: Students' Perception
  - Evaluation: Improvement on Students' Performance
  - Comparison with Related Works
- Similar approach adopted for creating **tutorial materials**:

Chen-Wei Wang. ***Integrating Drawing Tablet and Video Capturing/Sharing to Create Tutorial Materials***. In *14th International Conference on Computer Science and Education (**ICCSE**)*. IEEE, 2019.

Questions?

Teaching Challenge: Big Classes

Adopting the Approach



Evaluation: Student Perception (1)

Students answered anonymously on a 7-point scale:

1. The course helped me grow intellectually.
2. The course learning outcomes were clearly stated and achieved in the course.
3. The instructor conveyed the subject matter in a clear and well-organized manner.
4. The instructor helped me understand the importance and significance of the course content.
5. Overall, the instructor was an effective teacher in this course.

Evaluation: Student Perception (2)

COURSE	CS1	CS2	CS3
RESPONSE	58.09% (219/377)	58.42% (59/101)	85.73% (70/82)

		Q1	Q2	Q3	Q4	Q5
CS1	agree	82.33	90.6	not available		
	neutral	9.02	4.51	not available		
	disagree	7.15	4.14	not available		
CS2	agree	91.53	98.3	100	98.3	96.61
	neutral	6.78	0	0	0	1.69
	disagree	1.69	1.69	0	1.69	1.69
CS3	agree	80	80	94.28	98.3	90
	neutral	1.43	11.43	2.86	0	2.86
	disagree	18.57	8.58	2.86	10.0	7.25

Evaluation: Improvement on Performance (1)

Student Performance Measure in Various ***Complex Ideas***:

1. Subcontracting (Contracts in Descendant Classes)
2. The Visitor Design Pattern
3. Genericity
4. Formal Verification (Proving Loop Correctness and Termination)
5. OOP (Inferring Classes/Attributes/Methods from a Tester)

Evaluation: Improvement on Performance (2)

Course	CS3 (SU15)	CS3 (F17)
Proposed Technique Adopted?	No	Yes
Class Size	49	80
Topic	**Student Average Scores**	
Subcontracting	51.63%	54.81%
Visitor Pattern	51.33%	58.33%
Genericity	63.27%	67.00%
Formal Verification of Software	63.62%	63.17%
Course	CS1 (SP17)	CS1 (W18)
Proposed Technique Adopted?	No	Yes
Class Size	38	190
Topic	**Student Average Scores**	
Object-Oriented Programming	42.97%	56.4%

Index (1)

Challenges of Undergraduate Teaching
How Would You Help this Upset Student?
Motivating Question
Contribution:
An Approach for Effective After-Class Learning
Example Lecture: Class Model
Example Lecture: Console Tester
Example Lecture: Console Output
Example Lecture: Q & A
A Pattern for Teaching Complex Ideas
Contribution:
An Approach for Effective After-Class Learning
Study Resources for Students (1)
Study Resources for Students (2)

Index (2)

Teaching Context
Reflections
Beyond this talk...
Teaching Challenge: Big Classes
Adopting the Approach
Evaluation: Student Perception (1)
Evaluation: Student Perception (2)
Evaluation: Improvement on Performance (1)
Evaluation: Improvement on Performance (2)