Initiation à la pensée computationnelle avec JavaScript : le Cours STIC I

Mattia A. Fritz et Daniel K. Schneider

TECFA, Université de Genève, Faculté de Psychologie et Sciences de l'Éducation
42 bd Pont-d'Arve, CH-1211 Genève
mattia.fritz@unige.ch
daniel.schneider@unige.ch

Résumé. Cette contribution présente un retour d'expérience sur l'organisation d'un cours de Master en technologies éducatives visant à initier les étudiants à la pensée computationnelle à travers la conception, le développement et le déploiement de petites applications web interactives. Inséré dans un contexte de formation hybride, avec des longues périodes à distance, le cours s'appuie sur un dispositif pédagogique basé sur MediaWiki, à travers lequel les étudiants – ainsi que toute personne intéressée – ont accès et peuvent contribuer au matériel pédagogique. Ce matériel est composé par une collection de ressources théoriques, techniques et pratiques qui forment une initiation à la pensée computationnelle avec JavaScript.

Mots-clés. Pensée computationnelle, Interactivité, Computation, JavaScript, Développement Web

Abstract. This contribution presents the organization of a master course in educational technology aiming to introduce students to computational thinking through the design, development and deployment of small web-based applications. The course takes place in a blended format, with long periods of remote working, and uses a learning environment based upon a MediaWiki, through which learners – and all interested people – can access and contribute to the learning material. The material comprises a collection of theoretical, technical and practical resources, which have been bundled in an initiation to computational thinking with JavaScript.

Keywords. Computational Thinking, Interactivity, Computation, JavaScript, Web development

1 Introduction

Dans cette contribution, nous illustrons et analysons un cours universitaire appelé Sciences et Technologies de l'Information et de la Communication I (STIC I par la suite), proposé dans le cadre du *Master of Sciences in Learning and Teaching Technologies* (MALTT) à l'Université de Genève, dans la Faculté de Psychologie et des Sciences de l'Éducation. L'objectif du cours est d'initier les étudiants à la pensée computationnelle – que nous définissions comme un ensemble d'habiletés,

compétences et procédures qui facilitent la résolution de problèmes en s'appuyant sur des principes tirés des sciences informatiques [5] – en synergie avec le contexte plus général du Master : l'intégration des technologies éducatives dans l'apprentissage et la formation. L'initiation est considérée achevée si, à la fin du cours, (i) l'apprenant arrive, au niveau conceptuel, à se construire des représentations mentales du fonctionnement des agents computationnels au sens large ; et (ii) l'apprenant arrive, au niveau procédural, à poser une (simple) problématique, trouver une (simple) solution algorithmique, et l'implémenter dans une (simple) application web interactive.

Basé sur la pédagogie par projet et inséré dans un contexte de formation hybride prévoyant des longues périodes de travail à distance, le cours s'articule sur trois composantes que nous allons illustrer plus en détail dans cet article : (i) le matériel pédagogique de support [3] disponible, sous licence *Creative Commons,* à travers un wiki public que les étudiants peuvent non seulement consulter, mais également contribuer à améliorer ; (ii) des exemples et exercices disponibles, sous licence MIT, à travers la plateforme GitHub ; et (iii) le développement et déploiement de petites applications interactives en HTML, CSS et JavaScript que les étudiants peuvent concevoir et implémenter selon leurs intérêts et finalités souhaitées.

STIC I est suivi par des étudiants avec des profils académiques et professionnels très hétérogènes, mais souvent avec des connaissances préalables assez faibles d'un point de vue technique (voir plus bas). Aucun prérequis technique particulier n'est nécessaire, si ce n'est pour une connaissance de base de HTML et CSS qui, dans le cas du Master, est fournie dans un workshop de 2 jours. De ce fait, l'organisation du cours et le matériel pédagogique d'accompagnement pourraient intéresser un plus large public, que ce soit dans le cadre d'une formation ou en autoapprentissage.

L'article proposera d'abord un bref survol de l'évolution du cours dans ces dernières années, notamment en ce qui concerne l'abandon de Macromedia/Adobe Flash en faveur des la triade HTML5, CSS et JavaScript « pure ». Ensuite, nous illustrerons brièvement l'organisation du cours. Une partie de l'article sera alors consacrée à la description du matériel pédagogique. En guise de discussion, nous fournirons une synthèse des feedbacks des étudiants et des enjeux pour le cours.

2 Brève histoire de STIC I

Le Master est proposé depuis les années 1990 et le cours STIC I s'est toujours occupé de sensibiliser les étudiants sur les aspects techniques des technologies éducatives. Bien que la philosophie et l'approche du cours soient restées plutôt stables dans le temps, les aspects plus pragmatiques ont forcément évolué en fonction des technologies disponibles et des *trends* dans leur utilisation.

Ces dernières années en particulier ont été caractérisées par la nécessité de trouver une alternative à l'utilisation de Macromedia/Adobe Flash. En effet, cet environnement de développement a été utilisé pendant plusieurs années principalement en raison de la possibilité de combiner le multimédia et la programmation dans un même outil auteur relativement simple à utiliser. À la suite du déclin de Flash – causé principalement par la décision de ne plus supporter le plugin sur les dispositifs Apple et non pas pour des raisons pédagogiques – nous avons évalué deux options : (a) poursuivre l'utilisation de

l'environnement Adobe (e.g. Animate CC) basé sur les technologies HTML, CSS et JavaScript ; ou (b) passer directement à ces mêmes technologies, sans l'intermédiaire d'un outil auteur.

Ces dernières années étant également caractérisées par l'essor de la pensée computationnelle en éducation [2][4], nous avons décidé de mettre l'accent encore plus sur la programmation, tout en essayant de maintenir la partie multimédia intacte. De ce fait, nous avons opté pour une approche plus fondamentale au développement de pages web dynamiques. Ce choix permet de poursuivre deux objectifs : (i) sur le plan conceptuel, sensibiliser les étudiants au fonctionnement des agents computationnel à l'aide d'un langage de programmation [1] ; et (ii) sur le plan procédural, initier les étudiants à la pensée computationnelle à travers le développement d'applications web interactives [4].

En même temps, cette approche expose les apprenants à toutes les difficultés que les novices rencontrent lorsqu'ils sont confrontés aux mécanismes de la programmation [1][2][4] tels que la pensée symbolique, la portée des variables et des fonctions, la récursivité, etc. D'autant plus que la formation se déroule en large partie à distance, il était nécessaire de fournir aux étudiants des repères pour éviter de se perdre dans les labyrinthes du codage. Pour cette raison, nous avons investi dans l'amélioration du matériel pédagogique d'accompagnement qui sera présenté plus bas dans l'article.

3 Description du cours[1]

Le cours se déroule sur un semestre universitaire et contribue à l'acquisition de 6 crédits ECTS. Aucun prérequis technique particulier n'est nécessaire, si ce n'est pour une brève sensibilisation à HTML et CSS qui, dans le cas du Master, est faite en deux jours dans un atelier de préparation. En raison du format hybride du Master, le cours est divisé en 3 périodes caractérisées par une journée entière de cours en présence suivie par 4-5 semaines de travail en autonomie à distance. Les journées en présence sont organisées en *workshop* dans lesquels des parties théoriques sont alternées avec des démos et des activités *hands-on*. L'organisation du cours est gérée à travers des pages wiki[2] qui servent à (i) illustrer les informations et ressources pour les activités en salle de classe ; (ii) définir les consignes pour les projets à rendre ; et (iii) faciliter le support à distance à l'aide de l'onglet discussion, où les étudiants peuvent poser leurs questions.

Les six projets à rendre pour le cours explorent différents concepts liés aux technologies du web : (1) identité digitale, avec la création d'une page personnelle en HTML/CSS ; (2) création d'une *flash card* avec JavaScript ; (3) application interactive qui s'appuie sur un algorithme aléatoire, conditionnel ou itératif ; (4) interactivité des vidéos avec HTML5 et JavaScript ; (5) glisser-poser avec jQuery UI ; et (6) *framework* CSS et bibliothèques JavaScript. Chaque projet est constitué par : (a) la conception, le développement et le déploiement d'une (petite) application web interactive ; et (b) un rapport qui explique principalement les objectifs pédagogiques de l'application ainsi que le design du dispositif. Des contraintes assez spécifiques sont fournies au niveau

[1] Voir https://edutechwiki.unige.ch/fr/STIC:STIC_I_(Yoshi) pour une description plus détaillée du cours de l'année académique 2018/2019 auquel cet article fait référence

[2] Voir https://edutechwiki.unige.ch/fr/STIC:STIC_I_-_exercice_5_(Yoshi) pour un exemple

des caractéristiques que le dispositif doit présenter (p. ex., au moins un élément interactif, l'utilisation d'une structure de contrôle, la génération d'un élément aléatoire, etc.) ; mais au-delà de ces contraintes techniques, les étudiants sont libres de créer des applications sur des sujets et pour des publics cibles en fonction de leurs intérêts, connaissances préalables et parcours envisagés. Chaque production est téléversée par les étudiants sur un serveur web et rendue disponible sous forme d'un e-portfolio public[3].

4 Matériel pédagogique de support

Les activités en salle de classe étant très denses, les étudiants – surtout novices en programmation – sont débordés d'informations qu'ils oublient dans les jours suivants. Pour cette raison, nous avons décidé d'investir dans l'amélioration du matériel pédagogique qui accompagne les étudiants pendant les périodes à distance. Ce matériel a pris la forme d'une collection de ressources théoriques, techniques et pratiques que nous avons appelée « Initiation à la pensée computationnelle avec JavaScript » [3]. Dans cette perspective, JavaScript est utilisé en tant qu'*outil cognitif* qui permet d'explorer à la fois les aspects computationnels (variables, fonctions, …) et interactifs (DOM, gestionnaire d'événements, …) des pages web dynamiques. Le matériel combine deux types de ressources que nous allons illustrer plus en détail par la suite : (i) des pages wiki libres en accès et modifiables par les inscrits ; et (ii) des exemples et exercices disponibles au téléchargement à travers la plateforme GitHub.

4.1 Pages EduTech Wiki

Depuis 2006, TECFA héberge et maintient EduTech Wiki[4], un « MediaWiki[5] » dédié aux technologies éducatives conçu à la fois comme portail de partage de connaissances et environnement pédagogique sur le principe du « écrire pour apprendre » [6] et du constructivisme communautaire [7]. Libre en lecture, le wiki peut être modifié par tous les inscrits (l'inscription est ouverte au public, mais doit être validée par les administrateurs à fin d'éviter le spam). Nous avons décidé de baser le matériel pédagogique principalement sur EduTech Wiki pour deux raisons : (i) la philosophie ouverte, participative et évolutive des ressources wiki ; et (ii) la nature en prévalence textuelle du matériel.

Grâce à la philosophie ouverte, participative et évolutive, toute personne inscrite peut contribuer au matériel, ce qui est particulièrement utile dans une perspective de maintien et mise à jour. Les étudiants eux-mêmes ont contribué de manière conséquente à l'amélioration du matériel tel qu'il existe aujourd'hui, souvent avec des corrections orthographiques, mais également avec des commentaires et propositions. Une extension MediaWiki permet d'organiser les pages en modules, avec des menus de navigation qui facilitent le passage d'une ressource à l'autre (voir Fig. 1). Actuellement,

[3] Voir par exemple http://tecfaetu.unige.ch/etu-maltt/yoshi/lettry0/
[4] Voir https://edutechwiki.unige.ch/fr/Accueil
[5] Voir https://www.mediawiki.org/wiki/MediaWiki

trois modules – conçus pour des novices en programmation – sont utilisés dans le cadre de STIC I : « aspects théoriques de la pensée computationnelle », « concepts de base de JavaScript », et « JavaScript dans le navigateur ».

L'utilisation du wiki a également une raison pédagogique : privilégier une modalité en prévalence textuelle du matériel de support, notamment en contraste avec l'usage des vidéos. Bien que nous recevions régulièrement des requêtes d'enregistrer les séances en présence, nous essayons de décourager une approche basée sur la répétition des manipulations effectuées à l'écran. Le format en prévalence textuelle du matériel facilite le repère des informations ponctuelles, comme par exemple l'explication d'un morceau de code ou d'une technique spécifique. De plus, cette approche oblige les étudiants à lire du matériel ou de la documentation technique, ce qui représente l'un des « *soft skills* » principaux visés par le cours.

Fig. 1 Détail d'une page EduTech Wiki avec le menu de navigation sur la droite qui permet de passer facilement entre une ressource et l'autre, ou entre un module et l'autre.

4.2 Exemples et exercices de codage

La plupart des ressources de l'initiation à la pensée computationnelle avec JavaScript disponibles dans EduTech Wiki sont accompagnées par un *dépôt* GitHub qui contient les exemples de code illustrés dans la page, ainsi que des exercices de consolidation. Le choix de GitHub est encore une fois lié à la philosophie ouverte, participative et évolutive du matériel, car toute personne intéressée peut utiliser le matériel ou proposer des modifications à travers des *pull requests*. Ce mécanisme étant plus complexe par rapport à la modification d'une page wiki, pour l'instant aucune *pull request* a été faite par les étudiants (qui reçoivent une brève introduction à Git/GitHub).

Les exemples et exercices sont organisés afin que les étudiants puissent ouvrir les fichiers avec un éditeur de texte et s'exercer directement dans l'environnement de travail qu'ils utilisent ensuite pour développer leurs propres dispositifs. De ce fait, le transfert pédagogique relève non seulement du contenu des fichiers, mais également des procédures pour y accéder et pour les manier. Nous avons en effet remarqué que

ces opérations s'avèrent assez problématiques pour les débutants. Par conséquent, nous avons privilégié l'utilisation d'un éditeur de texte plutôt que des environnements intégrés comme JSFiddle ou CodePen, même si ces environnements sont brièvement abordés. De plus, les fichiers essayent de véhiculer des bonnes pratiques (p. ex., utilisation de commentaires, noms de fichiers sémantiques, …) et les exemples sont dans le thème des technologies éducatives (p. ex. génération aléatoire d'une lettre et un chiffre pour demander à l'utilisateur de nommer, par exemple, 4 mots initiant par F).

5 Evaluation

À la fin du semestre, les étudiants évaluent les différents cours du Master. En outre des items liés à la qualité de l'enseignement, l'évaluation de STIC I a porté également sur deux aspects d'intérêt pour cette contribution : (i) l'évaluation des connaissances préalables et acquises, ainsi que l'apport du court au niveau de la formation ou du parcours professionnel ; et (ii) l'évaluation du matériel pédagogique et des outils utilisés dans le cours. Onze étudiants (N=11) sur dix-huit inscrits au cours ont répondu, de manière anonyme, au questionnaire en ligne.

En ce qui concerne l'évaluation des connaissances, les étudiants partaient de connaissances préalables plutôt faibles, avec une moyenne de M=2.55 (SD=1.58, 95% CI [1.49, 3.60]) sur une échelle allant de 1 (très faibles) à 6 (très fortes). Après le cours, les connaissances acquises ont été évaluées comme étant plutôt fortes, avec une moyenne de M=4.81 (SD=0.75, 95% CI [4.31, 5.32]), ce qui correspond à une incrémentation moyenne de Md=2.27 (95% CI [1.23, 3.32], $t(10)=4.85$, $p < 0.001$ dans un test apparié bilatéral). L'apport du cours a été jugé fort en ce qui concerne le parcours de formation, avec une moyenne de M=5.00 (SD=1.26, 95% CI [4.14, 5.85]), mais également au niveau professionnel, avec une moyenne de M=5.09 (SD=1.04, 95% CI [4.39, 5.79]). Ces résultats (voir Fig. 2) semblent indiquer d'un côté la nécessité d'aborder des sujets très techniques, vu les faibles connaissances préalables, et de l'autre la pertinence de la programmation dans le parcours formatif et professionnel d'étudiants en technologies éducatives et domaines similaires.

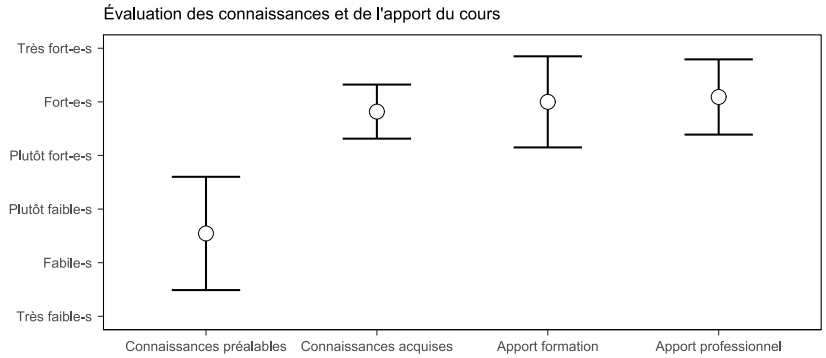

Fig. 2 Évaluation des connaissances avant et après le cours, ainsi que de l'apport du cours dans le parcours de formation et professionnel des étudiants (N=11). Les barres représentent les intervalles de confiance à 95%.

Nous avons aussi demandé aux étudiants leur degré d'accord sur l'utilité du dispositif pédagogique basé sur EduTech Wiki et les ressources complémentaires. Sur une échelle allant de 1 (pas du tout d'accord) à 6 (tout à fait d'accord), les étudiants ont trouvé la plateforme pédagogique utile, avec un degré d'accord moyen de M=5.45 (SD=1.04, 95% CI [4.76, 6.15]). De plus, ils ont trouvé l'usage d'un serveur web pour déployer leurs applications utile et pertinent, avec un degré d'accord moyen de M=5.91 (SD=0.09, 95% CI [5.71, 6.11]). Ils pensent pouvoir répliquer cette opération sur un autre serveur avec une certaine confiance, avec un degré d'accord moyen de M=5.27 (SD=0.14, 95% CI [4.96, 5.59]). Ces résultats (voir Fig. 3) semblent corroborer le choix d'une approche assez fondamentale au développement de pages web interactives et le déploiement à travers un serveur web « traditionnel » (c.-à-d., les étudiants utilisent un client SFTP pour téléverser leurs productions à des endroits précis sur le serveur).

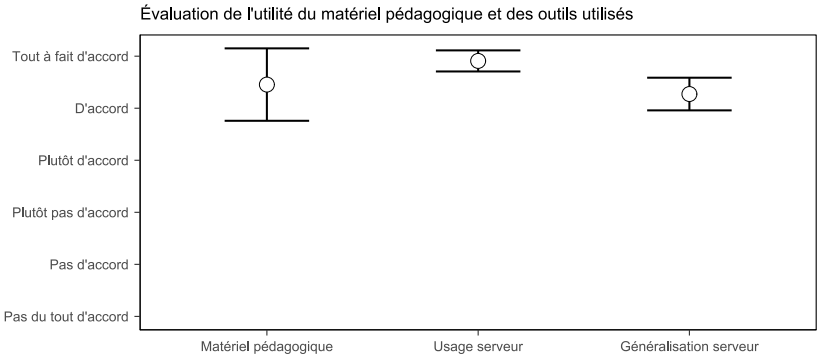

Fig. 3 Évaluation de l'utilité du matériel pédagogique et des outils utilisés pendant le cours (N=11). Les barres représentent les intervalles de confiance à 95%.

6 Discussion générale et conclusion

La structure et organisation du cours semblent favoriser les objectifs d'apprentissage et l'évaluation positive des étudiants est encourageante, notamment en fonction de la perception de son apport pour la formation et pour le futur professionnel des étudiants. De plus, le niveau initial du cours et la difficulté progressive paraissent adéquats au contexte : les étudiants partent de connaissances techniques initiales plutôt faibles, mais qu'ils perçoivent s'améliorer pendant le semestre. Cette progression est corroborée par le fait que pratiquement tous les étudiants arrivent en effet à rendre des dispositifs fonctionnels, ce qui représente l'objectif minimal pour réussir l'exercice. De plus, les dispositifs sont souvent orientés vers la résolution d'une problématique pédagogique bien identifiée dans le rapport, ce qui corrobore le choix d'une pédagogie par projet à travers laquelle les étudiants peuvent exprimer, à travers la programmation, leur créativité et capacité à implémenter un dispositif pédagogique qu'ils ont conçu, implémenté et déployé eux-mêmes.

En même temps, les solutions algorithmiques utilisées dans les dispositifs concernent en prévalence des aspects interactifs (p. ex. affichage/cacher des feedback, etc.). En revanche, les aspects computationnels restent souvent plus marginaux ou adaptés depuis des exemples en fonction des besoins spécifiques et dans une perspective orientée au résultat. En considération du caractère introductif du cours, les attentes à ce propos doivent forcément être pondérées sur la base des nombreuses difficultés que les étudiants rencontrent pour mener à bien leurs projets, d'autant plus que les applications envisagées au départ sont souvent trop ambitieuses par rapport au niveau technique et au temps à leur disposition pour rendre les travaux.

En outre, la pédagogie par projet ne permet pas d'évaluer si les étudiants ont intériorisé des concepts à propos du fonctionnement des agents computationnels. Le fait qu'ils arrivent à concevoir, développer et déployer des pages web dynamiques n'implique pas nécessairement une compréhension conceptuelle du fonctionnement des nombreux éléments sous-jacents, qu'il faudrait plutôt tester au niveau des connaissances déclaratives. Néanmoins, cette approche a l'avantage d'activer des compétences techniques transversales comme l'organisation de fichiers, l'utilisation d'un serveur distant pour publier des contenus sur le web, ou encore lire du matériel et de la documentation technique avec une application directe et immédiate.

Dans cette contribution, nous avons illustré les caractéristiques d'un cours universitaire dans un Master en technologies éducatives, dont l'objectif est d'initier des étudiants sans un background informatique à la pensée computationnelle. À travers cette initiation, les étudiants arrivent à concevoir, développer et déployer des petites applications web interactives, dont les objectifs pédagogiques sont décidés par les étudiants eux-mêmes. Pour atteindre cet objectif, les étudiants doivent utiliser la pensée computationnelle pour transformer leurs idées d'application dans des implémentations en HTML, CSS et JavaScript. Afin d'accompagner ce processus complexe, le cours prévoit l'utilisation de matériel pédagogique mis à disposition sous licence *Creative Commons* ou MIT. Grâce aux caractéristiques modulaires et à une philosophie ouverte, évolutive et participative, ce matériel peut être adapté et réutilisé dans des contextes pédagogiques formels ou en autoapprentissage.

References

1. Du Boulay, B., O'Shea, T., & Monk, J. (1981). Black box inside the glass box: Presenting computing concepts to novices. Int. J. Man-Machine Studies, 14(3), 237–249.
2. Fincher, S. A., & Robins, A. V. (Eds.). (2019). The Cambridge Handbook of Computing Education Research. Cambridge, UK: Cambridge University Press.
3. Fritz, M. A., & Schneider, D. K. (Eds.). (2018). Initiation à la pensée computationnelle avec JavaScript. EduTechWiki, TECFA, Université de Genève. Retrieved from https://edutechwiki.unige.ch/fr/pcjs
4. Guzdial, M. (2015). Learner-Centered Design of Computing Education. Research on Computing for Everyone. Morgan & Claypool.
5. Pensée computationnelle — EduTech Wiki. (n.d.). Retrieved March 22, 2019, from https://edutechwiki.unige.ch/fr/Pensée_computationnelle
6. Galbraith, D.: Cognitive models of writing. Ger. as a foreign Lang. 2–3, 7–22 (2009).
7. Holmes, B., Gardner, J.: E-learning: Concepts and practice. Sage (2006).