Copyright © 2020 Yves MARCOUX; dernière modification de cette page: 2020-03-11.

Historique des mises à jour :
2020-01-09 Mise en ligne initiale.
2020-01-11 Correction de trois pondérations à la section Évaluation.
2020-01-17 Certaines dates de la section Calendrier des activités étaient en erreur et ont été corrigées.
2020-01-24 L'emplacement du Quiz 2 dans la colonne centrale du Calendrier des activités a été corrigé.
2020-03-11 La colonne centrale du Calendrier des activités a été modifiée pour mieux refléter la progression réelle.
Les modifications sont présentées sur cette couleur de fond.

SCI6373 Programmation documentaire

Plan de cours Hiver 2020

Yves MARCOUX - EBSI - Université de Montréal

Table des matières

1. Informations générales
2. Objectifs
3. Description
4. Calendrier des activités
5. Évaluation
6. Règlements, politiques
   6.1 Remise des travaux à évaluer
   6.2 Modes de communication
   6.3 Autres règlements et politiques
7. Bibliographie
   7.1 Lectures obligatoires
   7.2 Ressources utiles et/ou amusantes
   7.3 Références diverses

---

1. Informations générales

- Cours de trois crédits, ce qui correspond à 135 heures de travail pour l'étudiant.
- **Description officielle du cours :**

  > Introduction à la programmation procédurale et orientée-objet. Développement d'applications documentaires dans un langage de scriptage orienté-objet. Aperçu d'autres langages. Notions d'algorithmique et de complexité. Préalables: SCI6051 et SCI6052.

- **Horaire et locaux:** Jeudi de 13h à 15h50 au C-2043 Pavillon Lionel-Groulx. Certains travaux de laboratoire s'effectuent à l'intérieur de la même plage horaire et au même endroit.
- **Professeur:** Yves MARCOUX <ymarcoux@gmail.com>.
- **Site du cours:** <https://cours.ebsi.umontreal.ca/SCI6373/>.

---

2. Objectifs

À la fin du cours, l'étudiant(e) possédera les rudiments de la programmation procédurale et orientée-objet moderne. Plus spécifiquement, l'étudiant(e):

- Comprendra le concept de scriptage en général et dans le contexte du web en particulier.
- Comprendra l'articulation client-serveur(s) du point de vue programmatique.
- Pourra développer à partir de zéro des scripts pour des traitements simples.
- Pourra adapter des scripts complexes existants.

---

3. Description

Le cours est une introduction à la programmation dans les environnements de scriptage. Le principal langage utilisé est le JavaScript. Un projet de fin de session permet d'intégrer les connaissances acquises. Les étudiants sont aussi appelés à explorer un autre langage ou environnement de programmation (de type scriptage ou non).

L'accent est mis sur le scriptage de pages web pour exécution en navigateur côté client, mais l'exécution côté serveur sera aussi touchée (si possible avec accès à une base de données), de même que le scriptage au niveau du système d'exploitation pour la manipulation de fichiers en lot. Une connaissance de base de HTML et de CSS est présupposée.

Plusieurs exercices sont donnés au cours de la session, en plus de devoirs évalués. Il est **essentiel** de faire les exercices proposés au fur et à mesure de façon à ne pas « décrocher ».

Le langage ou environnement choisi pour exploration doit être approuvé par le professeur suivant un protocole qui sera présenté en classe. Ce peut être une occasion d'explorer dans des projets amusants les ressources programmables des laboratoires d'informatique documentaire de l'EBSI, comme le `micro:bit`, le `Raspberry Pi` et le `Arduino`.

Le projet de fin de session est un devoir évalué d'un peu plus grande envergure que les autres.

---

4. Calendrier des activités

Notes:

- *L'heure limite de remise de certains travaux est fixée au **lundi à 23h55**. Comparativement à une remise en début de cours, cette formule offre la chance de consulter le professeur une fois de plus avant une remise.*
- *Les contenus, incluant le travail supervisé en laboratoire, et leur répartition entre les cours seront constamment ajustés au cours de la session.*

Calendrier des activités

Date – Cours	Contenu et lectures	Évaluations
2020-01-09 – C1	Bienvenue. Concepts de scriptage et de « programmation documentaire ». Programmation et société. « End-user programming » et « end-user development ». Programmation et écriture. Notions d'environnement et d'objet. Les environnements touchés dans le cours. JavaScript et navigateurs web : console JavaScript, pseudo-protocole « javascript: », « bookmarklets » et scripts imbriqués. Un premier script.	

2020-01-16 – C2	**Lectures préalables:** Une brève introduction au JavaScript; texte de Kolawole; Goodman, chap. 1 et 8. ***Facultatif :*** **Suggestion pour rafraîchir vos connaissances sur HTML et CSS :** Introduction à XHTML et CSS. Retour sur la lecture Une brève introduction au JavaScript, arbres d'exécution.	
2020-01-23 – C3	**Lectures préalables:** Goodman, chap. 9, 10, 15 et 16. Entrées-sorties de base. Enchaînement d'instructions, construction d'un script, méthode de travail, commentaires. Notion d'« approche algorithmique ». Présentation du Devoir 0.	
2020-01-30 – C4	**Lectures préalables:** Goodman, chap. 18, 21 (jusqu'à p. 396) et 22. Quiz 1 (en classe). Conversions, opérations sur les chaînes de caractères. Rappels HTML et CSS. Dynamisme dans une page HTML par stylage. Entrées-sorties par formulaire.	**13h :** Quiz 1 (10%)
2020-02-06 – C5	Traitement conditionnel (IF) et structures de contrôle. Organigrammes. Présentation du Devoir 1.	**Lundi 2020-02-10 23h55 :** Devoir 0 à rendre sur StudiUM (10%)
2020-02-13 – C6	Débogage avec les outils de développement en navigateurs.	
2020-02-20 – C7	Fonctions définies par le programmeur. Dynamisme par propriétés CSS.	
2020-02-27 – C8	Tableaux.	**Lundi 2020-03-02 23h55 :** Devoir 1 à rendre sur StudiUM (10%)
2020-03-05	*Semaine de lecture (pas de cours)*	
2020-03-12 – C9	Quiz 2 (en classe). Comparaisons de chaînes et tri d'un tableau. Propriété `innerHTML`. Présentation du Devoir 2. Approche algorithmique et exercices pour le Devoir 2. Structures de données complexes. Objets arbitraires. JSON. La fonction `document.write` revisitée; fenêtres surgissantes (*pop-up*).	**13h :** Quiz 2 (10%)
2020-03-19 – C10	Dates. Désaccentuation.	**Lundi 2020-03-23 23h55 :** Devoir 2 à rendre sur StudiUM (10%)
2020-03-26 – C11	Présentation du Devoir 3. Approche algorithmique pour le Devoir 3 (suite).	

	Discussion du projet de fin de session.	
2020-04-02 – C12	Conférencier invité (à confirmer, date sujette à changement) Approche algorithmique pour le projet de fin de session. **Lectures préalables:** Les (3) articles de Morgan, Colford et Breeding; les (2) articles de Wikipedia.	**Lundi 2020-04-06 23h55 :** Devoir 3 à rendre sur StudiUM (15%)
2020-04-09 – C13	Approche algorithmique pour le projet de fin de session (suite). Si le temps le permet, choix parmi les thèmes suivants : • Traitements côté serveur et accès à une base de données. • Scriptage au niveau du système d'exploitation, accès aux fichiers locaux. • AJAX (Asynchronous JavaScript and XML). • Principe *Model-view-controller* (MVC). • Logiciel libre. • SaaS (Software as a Service). • Modèles de calcul, machines de Turing, concepts d'algorithmique et de complexité. • API, bibliothèques logicielles, « frameworks » JavaScript : JQuery, AngularJS, etc. • Compilation versus interprétation. • Analyse versus programmation. • Méthodologies: découpage, modularité, documentation. *Design patterns*. • Intelligence artificielle, réseaux de neurones artificiels, apprentissage profond. • Informatique et cryptographie quantiques. • *Bitcoin*, *Blockchains* et grands livres distribués (*distributed ledgers*).	
2020-04-16 – C14	Présentations (en classe) des langages ou environnements explorés par certaines équipes. Si le temps le permet, choix parmi les thèmes mentionnés sous C13.	**13h :** Présentation en classe (Prestation de la présentation + matériel déposé sur StudiUM : 15%)
2020-04-23 – C15	Présentations (en classe) des langages ou environnements explorés par certaines équipes. Si le temps le permet, choix parmi les thèmes mentionnés sous C13.	**13h :** Présentation en classe (Prestation de la présentation + matériel déposé sur StudiUM : 15%)
2020-04-30 (dernier jour de la session)		**Jeudi 2020-04-30 23h55 :** Matériel de présentation à déposer sur StudiUM

		Jeudi 2020-04-30 23h55 : Projet de fin de session à rendre sur StudiUM (20%)

---

5. Évaluation

Activité	Pondération
Deux quiz individuels à livres fermés en classe (2 x 10%)	20%
Quatre devoirs (10% + 10% + 10% + 15%)	45%
Projet de fin de session	20%
Exploration et présentation en classe d'un langage ou environnement de programmation: prestation et matériel présenté	15%

Toutes les activités évaluées se font **en équipe de deux**, sauf les quiz. Les devoirs sont décrits dans des protocoles rendus disponibles en temps opportun.

Il n'y a pas de rapport à rendre pour le travail d'exploration d'un langage ou environnement de programmation, mais vous devez remettre sur StudiUM une copie des diapositives et de tout autre matériel utilisé pour la présentation.

Les directives données dans les différents protocoles de travaux doivent être suivies et, pour les travaux évalués, leur respect est un critère d'évaluation.

La méthode de calcul de la note finale du cours est présentée au <http://marcoux.ebsi.umontreal.ca/enseign/calculs-NG.html>.

Pour vous aider à suivre l'évolution des résultats de vos évaluations au cours du trimestre, et notamment connaître les notes requises dans les évaluations à venir pour atteindre une cote donnée (en tenant compte ou non du seuil), un Évaluateur de réussite sera disponible peu après le début du trimestre. N'hésitez pas à l'utiliser.

---

6. Règlements, politiques

6.1 Remise des travaux à évaluer

Les artefacts à évaluer (fichiers, rapports, etc.) doivent être remis **en format numérique**. Sauf exception (mentionnée le cas échéant dans le protocole du travail) ils doivent être déposés à l'emplacement prévu sur StudiUM **en un seul fichier** (compressé si nécessaire), au plus tard à l'heure prévue pour la remise. Ce fichier unique doit être conforme aux directives données dans le protocole du travail concerné; en particulier, il doit être **nommé** conformément à ces directives.

Si la remise comporte un rapport (le cas échéant, cela est spécifié dans le protocole du travail), ce rapport doit être en format PDF, OpenOffice ou Word, et doit être formaté pour du papier 8,5 x 11 po. **Les rapports doivent impérativement être structurés en sections numérotées et paginés**.

Exceptionnellement, après entente avec le professeur, certains artefacts peuvent être remis en format papier.

Si vous ne respectez pas les directives ci-dessus, vous êtes considéré comme n'ayant pas remis le travail.

Vous devez *impérativement* conserver une copie de sécurité intégrale des artefacts remis, *au moins jusqu'à réception de leur correction*. Vous devez être prêt à transmettre rapidement sur demande au correcteur (par

courriel ou autrement) une copie du matériel remis originellement, en cas de problème de lecture ou autre; c'est votre responsabilité.

6.2 Modes de communication

Le mode de communication privilégié du professeur vers les étudiant(e)s est la section *Nouvelles concernant le cours* de la page d'accueil du cours. Vous pouvez, si désiré, vous abonner à ces nouvelles par le truchement du fil RSS associé; sinon, vous devez consulter la section *Nouvelles* très régulièrement (au moins une fois par jour).

Le professeur peut également utiliser *le courriel* pour joindre les étudiant(e)s. L'adresse utilisée pour vous joindre sera celle enregistrée dans votre profil TI; assurez-vous que cette adresse soit valide et fonctionnelle en tout temps (voyez notamment à ne pas laisser votre boîte aux lettres se remplir). Vous devez lire très régulièrement votre courriel (au moins une fois par jour).

Le mode de communication privilégié des étudiant(e)s vers le professeur est le courriel. L'adresse courriel du professeur est <ymarcoux@gmail.com>. **SVP, toujours inscrire la mention [SCI6373] (incluant les crochets) au début de la ligne sujet de votre message.**

6.3 Autres règlements et politiques

Tous les règlements, politiques et directives énoncés dans le Guide de l'étudiant de la MSI s'appliquent, incluant le *Code d'honneur de l'EBSI*. Une attention particulière est à porter aux éléments suivants :

Travaux en équipe
Pour les travaux réalisés en équipe, le professeur se réserve le droit d'évaluer séparément chaque membre d'une équipe.

Règlement disciplinaire sur le plagiat ou la fraude concernant les étudiants
Toute infraction au règlement sur le plagiat ou la fraude sera traitée suivant la procédure indiquée dans le règlement.

Retard dans la remise des travaux
Tout retard non justifié dans la remise d'un travail sera sanctionné : 5% de la note est retranché par jour de calendrier de retard jusqu'à un maximum de 35%; à la 8ième journée de calendrier, la note F ou zéro (0) est attribuée.

Qualité de la langue
Un maximum de 10% de la note globale d'un travail pourra être retranché pour mauvaise qualité de la langue dans les travaux (ne s'applique pas aux examens ou quiz en classe).

---

7. Bibliographie

7.1 Lectures obligatoires

- Breeding, Marshall. *ASIS&T Bulletin*, December 2008/January 2009. "The Viability of Open Source ILS." En ligne <https://asistdl.onlinelibrary.wiley.com/doi/full/10.1002/bult.2008.1720350207>.
- Colford, Scot. *ASIS&T Bulletin*, December 2008/January 2009 "Explaining Free and Open Source Software." En ligne <https://asistdl.onlinelibrary.wiley.com/doi/10.1002/bult.2008.1720350205>.
- Goodman, Danny; et al. *JavaScript Bible Seventh edition*. Indianapolis, Ind. : Wiley Pub., c2010, env. 2000 pp. En ligne sur ProQuest Ebook Central (de `umontreal.ca`) <http://atrium.umontreal.ca/UM:UM-ALEPH002244488>. Exemplaire papier de la 5e édition en réserve à la BLSH (QA 76.73 J39 G66 2004) <http://atrium.umontreal.ca/UM:UM-ALEPH000383396>.

- Kolawole, Emi. “Tim Berners-Lee on the making of new worlds.” *The Washington Post (Blogs) – Innovations*, 2013-03-11. En ligne <https://tinyurl.com/ygbnb9kj>.
- Marcoux, Yves. « Une brève introduction au JavaScript ». En ligne <https://cours.ebsi.umontreal.ca/SCI6373/H2020/matthem/introJS.html>.
- Morgan, Eric Lease. “Open Source Software in Libraries.” *ASIS&T Bulletin*, December 2008/January 2009. En ligne <https://asistdl.onlinelibrary.wiley.com/doi/10.1002/bult.2008.1720350204>.

7.2 Ressources utiles et/ou amusantes

- bento : Teach yourself to code. Site recensant un grand nombre de tutoriels pour apprendre à programmer dans différents langages. D’intérêt particulier est la grille <https://www.bento.io/topics>, dans laquelle les tutoriels sont regroupés par langage et par type de technologie. En ligne <https://www.bento.io/>.
- Broussard, Meredith. *Artificial unintelligence : how computers misunderstand the world.* Cambridge, Massachusetts : The MIT Press, 2018. En réserve à la BLSH (QA 76.9 C66 B76 2018).
- Cagle, Kurt. “Why JavaScript is the Future of Enterprise Computing.” *Avalon Consulting, LLC - Blogs*, 2013-02-18. En ligne <http://blogs.avalonconsult.com/blog/generic/why-javascript-is-the-future-of-enterprise-computing/>.
- Chandra, Vikram. *Geek Sublime : Une vision esthétique, littéraire, mathématique et pleine d’autodérision du codage.* Paris : Robert Laffont, 2014, 332 pp. Version anglaise *Geek Sublime: The Beauty of Code, the Code of Beauty.* En réserve à la BLSH (PN 56 T37 C36 2014) <https://atrium.umontreal.ca/UM:UM-ALEPH002208262>.
- Classements de langages de programmation selon différents critères. TIOBE, UBUNTUPIT, PYPL, etc.
- Code Studio. Site pour apprendre aux jeunes à programmer dans différents environnements attrayants. En ligne <https://code.org/>.
- Freeman, Jason; Brian Magerko; Doug Edwards; Tom Mcklin; Taneisha Lee; Roxanne Moore. 2019. EarSketch: engaging broad populations in computing through music. *Commun. ACM* 62, 9 (August 2019), 78–85. En ligne <https://doi.org/10.1145/3333613>.
- Krill, Paul. “Scripting Languages Spark New Programming Era.” *InfoWorld*, 2008-06-23. Résumé en ligne <https://technews.acm.org/archives.cfm?fo=2008-06-jun/jun-25-2008.html&hdr=1#367980>. Article complet au <https://www.infoworld.com/article/2651797/application-development/scripting-languages-spark-new-programming-era.html>.
- Manfre, Charles. “Why JavaScript is the Future of Programming.” *sitepoint - JavaScript*, 2012-11-30. En ligne <http://www.sitepoint.com/why-javascript-is-the-future-of-programming/>.
- Marji, Majed. *Le grand livre de Scratch.* Paris : Eyrolles 2017. En réserve ÉPC-Biologie - Didacthèque Jeunesse (005.13 M3444g 2017).
- Morgan, Nick. *JavaScript for Kids: A Playful Introduction to Programming.* Los Angeles (É.-U.) : No Starch Press, 2015, 305 pp. Version française à l’UdeM : *Javascript pour les kids.* Paris : Eyrolles 2017. En réserve ÉPC-Biologie - Didacthèque Jeunesse (006.77 M8494j 2017).
- Mozilla Developer Network. « Référence JavaScript. » En ligne <https://developer.mozilla.org/fr/docs/Web/JavaScript/>.
- Scratch. Langage de programmation graphique pour apprendre aux jeunes à programmer dans un environnement attrayant. En ligne <https://scratch.mit.edu/>.
- Stavely, Allan M. *Writing in software development.* New Mexico Tech Press, 2011. En réserve à la BLSH (QA 76.76 D63 S73 2011) <https://atrium.umontreal.ca/UM:UM-ALEPH002013882>.
- W3Schools. *JavaScript Tutorial.* En ligne <https://www.w3schools.com/js/>.
- Wikipedia. “Ajax (programming).” En ligne <http://en.wikipedia.org/wiki/AJAX>.
- Wikipedia. “Model-view-controller.” En ligne <http://en.wikipedia.org/wiki/Model-view-controller>.
- Wikipedia. “Software as a service.” En ligne <https://en.wikipedia.org/wiki/Software_as_a_service>. En particulier, la section *Criticism*.
- Wright, Nolan. “Why JavaScript Will Become The Dominant Programming Language Of The Enterprise.” *readwrite*, 2013-08-09. En ligne <https://readwrite.com/2013/08/09/why-javascript-will-become-the-dominant-programming-language-of-the-enterprise>.

7.3 Références diverses

- Babcock, Charles. “Ajax 101: From Toolkits To Strategy, How Companies Can Put It To Use.” InformationWeek, 2006-06-12. En ligne <https://www.informationweek.com/ajax-101-from-toolkits-to-strategy-how-companies-can-put-it-to-use/d/d-id/1044053>.
- Ecma International. *ECMAScript® 2019 Language Specification*, Standard ECMA-262, 10th Edition, June 2019. En ligne (PDF et HTML) <https://www.ecma-international.org/publications/standards/Ecma-262.htm>.

  C’est la spécification ultime du JavaScript, mise à jour pratiquement à chaque année. Malheureusement, illisible pour les non spécialistes.

- Festa, Paul. “Standards Activists Target Scripts.” *CNET News* 2005-07-19. En ligne <https://www.cnet.com/news/standards-activists-target-scripts/>. Résumé en ligne <http://technews.acm.org/archives.cfm?fo=2005-07-jul/jul-22-2005.html&hdr=2#item14>.
- Festa, Paul. “Will Ajax help Google clean up?” *CNET News*, 2007-06-21. En ligne <https://www.cnet.com/news/will-ajax-help-google-clean-up/>.
- Kaneshige, Tom. “A future without programming.” *InfoWorld*, 2008-11-20. En ligne <https://www.infoworld.com/article/2654154/application-development/a-future-without-programming.html>.
- Krill, Paul. “JavaScript creator ponders past, future.” *InfoWorld*, 2008-06-23. En ligne <https://www.infoworld.com/article/2653798/application-development/javascript-creator-ponders-past--future.html>.
- Kumar, Ritesh. “On Scripting Languages” *OSnews*, 2004-05-13. En ligne <https://www.osnews.com/story/7038>.
- Microsoft Corp. “Windows Script Host.” 2015-05-07. En ligne <https://msdn.microsoft.com/en-us/library/9bbdkx3k(v=vs.84).aspx>.
- Roque, Celine. “What Will It Take to Be a Web Worker in 2009?” *GIGAOM Research*, 2008-12-22. En ligne <https://gigaom.com/2008/12/22/what-will-it-take-to-be-a-web-worker-in-2009/>.
- Savill, John. “How do I run a windows script from the command line?” WindowsITPro. En ligne <https://www.itprotoday.com/management-mobility/how-do-i-run-windows-script-command-line>.

W3C XHTML 1.0 W3C CSS