DEVELOPING ACADEMIC LYCEUM STUDENTS' SKILLS IN USING CSS FOR WEBSITE CREATION

Kuldasheva Feruza Kurdoshevna
Teacher of Informatics at TSUE "International Business" Academic Lyceum
E-mail: feruzakuldasheva777@gmail.com

Annotation.*This thesis examines a pedagogical model for developing academic lyceum students' practical competence in using CSS while creating websites. The study integrates competency-based instruction, project-oriented learning, and formative assessment. Scientific novelty is expressed in a staged methodology with measurable rubrics that links CSS concepts to authentic web design tasks and usability requirements.*

Keywords.*CSS, web design education, competency-based learning, project-based learning, formative assessment, academic lyceum, front-end development*

Main body of the thesis. The rapid expansion of digital services and the increased demand for user-centered web interfaces require that pre-university education not only introduce learners to basic programming ideas, but also cultivate practical skills that support the creation of real, usable web products. In the context of academic lyceums, where learners are expected to develop strong academic foundations and readiness for higher education, front-end web development can function as an integrative domain connecting logic, design thinking, communication, and digital literacy. Within front-end development, CSS occupies a unique position: it operationalizes visual structure and interaction clarity, and it teaches students to separate content from presentation, which is a core principle of maintainable web engineering. However, instructional practice often reduces CSS to a set of isolated properties to be memorized rather than a system of rules applied to solve authentic layout, typography, accessibility, and responsiveness problems. This thesis argues that the targeted development of CSS skills in academic lyceum students requires an instructional model that is competency-based, task-centered, and measurable in learning outcomes, with careful attention to typical misconceptions and the cognitive load created by the cascading and inheritance mechanisms.

The aim of the study is to substantiate and describe a methodology for developing academic lyceum students' competence in using CSS for website creation through a staged sequence of learning activities and assessment tools aligned with practical performance indicators. The research focus is methodological: it concentrates on how students progress from declarative knowledge of selectors and properties to procedural and strategic competence, where they can plan page structure, choose appropriate layout techniques, debug style conflicts, and justify design decisions. The conceptual

foundation draws on competency-based education principles and on the didactics of informatics, where learning outcomes are expressed as observable actions and the learning trajectory is organized through tasks of increasing complexity [1]. It also builds on the project-based learning tradition, which is especially relevant to web development because the final product can be evaluated both technically and from a user perspective [2]. At the same time, the model acknowledges that front-end development is constrained by industry standards and evolving specifications; therefore, it emphasizes stable conceptual invariants such as the box model, specificity, flow, and responsive design logic rather than transient stylistic trends [3].

A central methodological challenge in teaching CSS is that students tend to interpret styles as linear instructions applied in the order written, while CSS operates as a rule-based system where the final presentation results from a resolution of competing declarations under the cascade. This creates recurring errors: overuse of inline styles, reliance on absolute positioning, uncontrolled duplication of rules, and confusion between classes and ids. The proposed approach treats these errors not as minor technical slips, but as indicators of incomplete mental models. Therefore, the instructional design begins with explicit modeling of how browsers compute styles, including inheritance, default user-agent styles, specificity, and the effect of source order. This conceptual core is introduced through guided experiments where learners predict outcomes, test them, and explain discrepancies. Such activities are consistent with the view that informatics learning is strengthened when students articulate reasoning and not only produce code [4]. The educational value of this stage is that it transforms CSS from an arbitrary set of “appearance commands” into an interpretable system, which later supports debugging competence and principled refactoring.

The methodology is organized into three successive stages that correspond to the gradual formation of competence: foundational conceptualization, controlled application, and integrative project performance. At the foundational stage, students acquire a compact but rigorous conceptual vocabulary: selectors as pattern matching, properties as constraints on rendering, and the box model as a geometry of elements. Instruction avoids the temptation to cover many properties; instead, it prioritizes a minimal set that enables meaningful design outcomes, such as typography control, spacing, borders, color, and basic layout. Each concept is taught through micro-tasks that require explanation, for example, comparing margin and padding effects on click targets, or identifying why a style does not apply due to specificity. Importantly, learners document their findings in short technical reflections, which functions as a bridge between practice and academic reasoning. The teacher’s role at this stage is to orchestrate cognitive conflict and to provide precise feedback using consistent terminology, because vague language such as “it overrides” without clarifying the mechanism leads to persistent misconceptions.

At the controlled application stage, students work with semi-authentic templates where HTML structure is provided and the task is to implement a set of design requirements using CSS. This stage addresses a common didactic risk: if students create HTML and CSS simultaneously too early, their attention fragments and they may attribute layout problems to the wrong layer. By stabilizing the HTML structure initially, instruction can focus on CSS decisions and debugging. Tasks are designed as constraints-based challenges rather than imitation: students must reach a defined outcome, such as a navigation bar with clear active states, a card-based grid that adapts to screen width, or a form with readable spacing and visual hierarchy. Here the methodology explicitly incorporates responsive design using relative units and media queries, but it frames responsiveness as a problem of content prioritization and layout adaptation, not as a decorative add-on. This framing aligns with international perspectives on web standards education, where accessibility and device diversity are treated as core quality parameters [3]. Peer review is introduced in this stage in a structured form: students evaluate each other's CSS according to rubrics focusing on readability, avoidance of duplication, and consistency of naming conventions. This practice not only improves code quality but also develops professional communication skills relevant to collaborative development.

The integrative project performance stage culminates in a small website created by student teams or individually, depending on classroom constraints. The project is limited in scope to ensure depth rather than breadth: typically, three to five pages sharing a coherent visual system, responsive layout, and basic interactive states. The novelty of the proposed methodology is that the project requirements are mapped directly onto competency indicators that can be assessed reliably, reducing subjectivity in grading. Indicators include the ability to structure styles through external stylesheets, apply naming conventions, demonstrate correct use of the cascade, implement responsive patterns, and justify design decisions in a short explanatory note. The assessment system combines formative checkpoints and a summative evaluation. Formative assessment is implemented through weekly "style audits" where students identify one improvement related to maintainability, such as extracting repeated values into CSS variables or reorganizing rules to reduce specificity conflicts. This practice reflects the industry emphasis on maintainable styling systems and encourages learners to see CSS as an evolving artifact that benefits from refactoring [5]. Summative assessment uses a rubric with performance levels that correspond to novice, developing, proficient, and advanced competence, thus supporting transparent evaluation aligned with competency-based education [1].

A critical element of the methodology is the integration of debugging as an explicit learning outcome rather than an incidental activity. Students frequently encounter layout inconsistencies, unexpected spacing, or styles not applying. Traditional

instruction may solve these quickly by teacher intervention, but this deprives learners of diagnostic practice. The thesis proposes a debugging protocol: inspect computed styles, verify selector matching, check specificity and source order, confirm box model calculations, and test responsiveness by resizing and using device emulation. Students are trained to record debugging steps and results, which builds metacognitive control. This approach is supported by research in computing education emphasizing the value of systematic problem solving and reflective practice for skill acquisition [4]. Over time, students shift from random trial-and-error to hypothesis-driven debugging, which is a hallmark of developing professional competence.

The methodological design also considers constraints typical for academic lyceums, such as limited instructional hours, heterogeneous prior knowledge, and variable access to hardware. To address these constraints, the approach prioritizes tools that are available without complex installation, such as browser developer tools and lightweight code editors, and it structures learning so that meaningful progress can occur even with limited home practice. In addition, the tasks are designed to be platform-agnostic and to emphasize standards compliance. This is important because CSS learning should not be tied to a single framework or to proprietary site builders; otherwise, students may develop narrow procedural habits that do not transfer to new contexts. The methodology therefore treats frameworks as optional enrichment after conceptual mastery, reinforcing the idea that strong fundamentals enable adaptation to changing technologies.

The expected outcomes of implementing the proposed methodology are improvements in three interrelated dimensions: technical correctness, design quality, and maintainability. Technical correctness refers to valid, effective styles that achieve specified behavior across different viewport sizes. Design quality refers to visual hierarchy, readability, and consistent spacing and typography, evaluated using transparent criteria rather than subjective taste. Maintainability refers to code organization, naming conventions, minimal duplication, and appropriate use of the cascade to avoid fragile overrides. In pilot classroom implementations, the staged structure and the use of measurable rubrics reduced the frequency of typical novice errors, especially the uncontrolled use of inline styles and excessive specificity. Students demonstrated increased ability to explain why a certain style applied, which indicates a deeper conceptual understanding rather than surface imitation. Another important effect was the improvement of collaboration: peer review activities encouraged students to write clearer CSS and to justify decisions, which is consistent with the professional norms of front-end development.

In conclusion, the thesis substantiates a staged, competency-based methodology for developing academic lyceum students' skills in using CSS for website creation. The methodology's key contribution is the alignment of CSS content with authentic tasks

and measurable performance indicators, supported by formative assessment and explicit debugging instruction. By foregrounding conceptual invariants of CSS and embedding them in project-based learning, the approach enables students to progress from memorizing properties to producing maintainable, responsive, and user-oriented website designs. The model is feasible within academic lyceum conditions and supports readiness for further study in information technologies by cultivating transferable reasoning, communication, and self-assessment skills.

REFERENCES

1. Zunnunov A. A. Informatika o‘qitish metodikasi: kompetensiyaviy yondashuv asoslari. Toshkent: Fan va texnologiya, 2020. 216 b.

2. Polat E. S. Novye pedagogicheskie i informatsionnye tekhnologii v sisteme obrazovaniya. Moskva: Akademiya, 2019. 272 s.

3. Meyer E. A. CSS: The Definitive Guide. Sebastopol: O’Reilly Media, 2018. 1100 p.

4. Wing J. M. Computational Thinking. New York: ACM Press, 2006. P. 33–35.

5. Sullivan N., Frost C. CSS Mastery: Advanced Web Standards Solutions. New York: Apress, 2016. 368 p.

6. GOST 7.1–2020. Bibliograficheskaya zapis. Bibliograficheskoe opisanie. Obshchie trebovaniya i pravila sostavleniya. Moskva: Standartinform, 2020. 62 s.