# Appendix A. Selected studies

Table 5. Selected studies

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Database** | **STUDY** | **App(s)** | **Study year** | **Publication year** |
| [1] | **ACM** | Flannery, L. P., Silverman, B., Kazakoff, E. R., Bers, M. U., Bontá, P., & Resnick, M. (2013). Designing ScratchJr: Support for early childhood learning through computer programming. In *Proceedings of the 12th international conference on interaction design and children* (pp. 1-10). | ScratchJr | 2013 | 2013 |
| [2] | **ACM** | Portelance, D. J., & Bers, M. U. (2015). Code and Tell: Assessing young children's learning of computational thinking using peer video interviews with ScratchJr. In *Proceedings of the 14th international conference on interaction design and children* (pp. 271-274). | ScratchJr | 2015 | 2015 |
| [3] | **ERIC** | Portelance, D. J., Strawhacker, A. L., & Bers, M. U. (2016). Constructing the ScratchJr programming language in the early childhood classroom. *International Journal of Technology and Design Education*, *26*(4), 489-504. | ScratchJr | 2014\* | 2016 |
| [4] | **ERIC** | Sullivan, A., Bers, M., & Pugnali, A. (2017). The impact of user interface on young children’s computational thinking. *Journal of Information Technology Education: Innovations in Practice*, *16*(1), 171-193. | ScratchJr | 2016\* | 2017 |
| [5] | **ERIC** | Strawhacker, A., Lee, M., & Bers, M. U. (2018). Teaching tools, teachers’ rules: exploring the impact of teaching styles on young children’s programming knowledge in ScratchJr. *International Journal of Technology and Design Education*, *28*(2), 347-376. | ScratchJr | 2014 | 2018 |
| [6] | **ERIC** | Strawhacker, A., & Bers, M. U. (2019). What they learn when they learn coding: investigating cognitive domains and computer programming knowledge in young children. *Educational Technology Research and Development*, *67*(3), 541-575. | ScratchJr | 2018\* | 2019 |
| [7] | **ERIC** | Chou, P. N. (2020). Using ScratchJr to Foster Young Children’s Computational Thinking Competence: A Case Study in a Third-Grade Computer Class. *Journal of Educational Computing Research*, *58*(3), 570-595. | ScratchJr | 2019\* | 2020 |
| [8] | **IEEE** | Thuzar, A., & Nay, A. (2015). Teaching and learning through creating games in ScratchJr: Who needs variables anyway!. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)* (pp. 139-141). IEEE. | ScratchJr | 2015 | 2015 |
| [9] | **LearnTechLib** | Rose, S., Habgood, J., & Jay, T. (2017). An exploration of the role of visual programming tools in the development of young children's computational thinking. *Electronic Journal of e-learning*, *15*(4), 297-309. | ScratchJr & Lightbot | 2016\* | 2017 |
| [10] | **Scholar** | Karadeniz, S., Samur, Y., & Özden, M. Y. (2014). Playing with algorithms to learn programming: A case study on 5 years old children. In *9th International Conference on Information Technology and Applications (ICITA2014)* (pp. 1-4). | Kodable | 2013\* | 2014 |
| [11] | **Scholar** | Kazakoff, E. R. (2015). Technology-based literacies for young children: Digital literacy through learning to code. In *Young Children and Families in the Information Age* (pp. 43-60). Springer, Dordrecht. | ScratchJr | 2014\* | 2015 |
| [12] | **Scholar** | Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using Scratch Jnr. On the iPad. *Journal of Computer Assisted Learning*, *32*(6), 576-593. | ScratchJr | 2015 | 2016 |
| [13] | **Scholar** | Falloon, G., Hale, P., & Fenemor, T. (2016). Planning and implementing coding in the junior classroom for competency and thinking-skill development. *Set: research information for teachers*, *2016*(1), 8-16. | ScratchJr & Daisy the Dinosaur | 2015 | 2016 |
| [14] | **Scholar** | Kalogiannakis, M., Ampartzaki, M., Papadakis, S., & Skaraki, E. (2018). Teaching natural science concepts to young children with mobile devices and hands-on activities. A case study. *International Journal of Teaching and Case Studies*, *9*(2), 171-183. | ScratchJr | 2017 | 2018 |
| [15] | **Scholar** | Lowe, T., & Brophy, S. (2019). Identifying Computational Thinking in Storytelling Literacy Activities with Scratch Jr. In *ASEE Annual Conference Proceedings* (p. 10). | ScratchJr | 2018 | 2019 |
| [16] | **Scholar** | Herheim, R., & Severina, E. (2020). Scratch programming and student’s explanations. *Mathematics Education in the Digital Age (MEDA)*, 45. | ScratchJr | 2019 | 2020 |
| [17] | **ScienceDirect** | Sheehan, K. J., Pila, S., Lauricella, A. R., & Wartella, E. A. (2019). Parent-child interaction and children's learning from a coding application. *Computers & Education*, *140*, 103601. | ScratchJr | 2018 | 2019 |
| [18] | **Scopus** | Papadakis, S., Kalogiannakis, M., & Zaranis, N. (2016). Developing fundamental programming concepts and computational thinking with ScratchJr in preschool education: a case study. *International Journal of Mobile Learning and Organisation*, *10*(3), 187-202. | ScratchJr | 2015 | 2016 |
| [19] | **Scopus** | Gomes, T. C. S., Falcão, T. P., & Tedesco, P. C. D. A. R. (2018). Exploring an approach based on digital games for teaching programming concepts to young children. *International Journal of Child-Computer Interaction*, *16*, 77-84. | Lightbot & Kodable | 2016 | 2018 |
| [20] | **Scopus** | Pila, S., Aladé, F., Sheehan, K. J., Lauricella, A. R., & Wartella, E. A. (2019). Learning to code via tablet applications: An evaluation of Daisy the Dinosaur and Kodable as learning tools for young children. *Computers & Education*, *128*, 52-62. | Daisy the Dinosaur & Kodable | 2016 | 2019 |
| [21] | **Scopus** | Govind, M., Relkin, E., & Bers, M. U. (2020). Engaging Children and Parents to Code Together Using the ScratchJr App. *Visitor Studies*, *23*(1), 46-65. | ScratchJr | 2018 | 2020 |

\* No exact info was mentioned within the paper regarding the date the study was implemented.

# Appendix B. Studies details

Table 6. Selected studies details

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Country** | **N** | **Children age / Grade level** | **Settings** | **Duration** | **Study design** | **Tasks** | **Assessment method** | **Analysis methods** | **CT or other skills assessed** |
| [1] | USA | 125 | 5-7 | Informal / Formal | Phase 1:  Pilot-test 1: 3-4 30' sessions  Pilot-test 2: Summer program ten h curriculum  Phase 2:  Nine sessions: 30-60' | Pilot trials | ScratchJr projects | Data collection /  project analysis | Baseline observations | ScratchJr programming function |
| [2] | USA | 66 | Second grade | Formal | 13 days /  1 hour per day | Interviews | Adapted version of the ScratchJr ‘Animated Genres’ curriculum | Artifact-based video interviews with students in pairs | Video analysis | ScratchJr is a tool for learning computational thinking |
| [3] | USA | 62 | Kindergarten –  2nd grade | Formal | Six weeks  One h / 2 per week | Quasi-experimental  (pre / posttest) | Scratch projects | Projects analysis | Data analysis | Basic CT skills |
| [4] | USA | 28 | 4-7 | Informal | Five days/ 3h per day | Mixed-method approach | ScratchJr projects | ScratchJr projects | PTD Framework | CT & students’ engagement |
| [5] | USA | 6 teachers / 222 children | Kindergarten –  2nd grade | Formal | 2-7 sessions (45') | Mixed-methods study | Solve It Tasks | ScratchJr Solve It assessments | Quantitative and qualitative data | Sequence, debugging, goal-oriented programming |
| [6] | USA | 57 | Kindergarten - 2nd grade | Formal | Six weeks,  twice-weekly 1-hour lessons /  12 classroom hours | Quantitative and qualitative case study | Lesson activities adapted from the Animated Genres curriculum | Post-intervention assessment | Analysis of children’s errors | ScratchJr language knowledge and underlying reasoning |
| [7] | Taiwan | 12 | 3rd grade | Formal | 8-week educational training (semester) | Quasi-experimental | Eight projects in class and take-home assignments | Competence test | Quantitative & qualitative analysis | CT competence |
| [8] | USA | N/A\* | K-2 graders | Informal | Half-day week-long | ScratchJr project | Games creation | ScratchJr projects | Code and tell | Designing & planning, sequencing, problem-solving |
| [9] | England | 40 | 6-7 | Formal | Each child played their version of the game for 30 minutes. | Exploratory approach | Children use both versions of the game. | A range of measures | Quantitative data | Both groups had a similar overall performance |
| [10] | Turkey | 25 | 5 | Formal | Three weeks | Case study | 5 lesson plans activities | Three worksheets | Data analysis | Sequence, conditional, and loops |
| [11] | USA | 54-60 | Kindergarten - combined first / 2nd grade | Formal | Eight weeks (1 hour daily / 2 days per week) | Quasi-experimental | Scratch programming projects | Data collection/project analysis | Baseline observations | Digital literacy skills |
| [12] | New Zealand | 32 | 5-6 | Formal | Five sessions /  25-40' | Mixed-method approach | Numeracy-focused topic (geometry) | Data collection/  project analysis | Video analysis | Coding tasks |
| [13] | New Zealand | 36 | Junior Primary (Year 1 and 2) | Formal | N/A | Case study | Geometry coding challenges | Interview, document analysis, iPad display, audio capture & a specially developed app | Quantitative and qualitative data | Daisy for learning the “mechanics” of building code. ScratchJr proved useful. Students need help |
| [14] | Greece | 17 | 5-7 | Informal | Two weeks /  1 hour per week | Pilot study | Digital scenarios | Digital scenarios | Quantitative and qualitative data | Basic concepts & principles of coding & CT |
| [15] | USA | 18 | 1st grade | Formal | Two half-day CT lessons | Qualitative analysis | 1st lesson: algorithms 2nd lesson: ScratchJr | Children's stories evaluation | Qualitative data | ScratchJr is a powerful accelerator for CT |
| [16] | Norway | N/A | 1st-year pre-service teachers and 4-year students | Formal | Two mathematics lessons /  Two days | Design-based research project | Students used multiplication tables to navigate characters in ScratchJr | ScratchJr projects | Audio & video recordings | ScratchJr can foster math concepts |
| [17] | USA | 31 parents /  31 children | 4-5 | Informal | Parent-child dyad lab visits.  Time spend N/A\*. | Parent-child interaction coding using a coding scheme | Free play with the PBS KIDS ScratchJr | Audio or video session transcribed & coded | Video-audio / analysis | Young children can learn coding skills from the ScratchJr app |
| [18] | Greece | 43 | Kindergarten | Formal | 13 hours  (1 hour daily / twice weekly) | Qualitative and quantitative analysis | ScratchJr projects | Project analysis for CT skills | Data analysis | Basic CT skills |
| [19] | Brazil | 42 | 5-7 | Formal | Nine weekly  40- minute meetings | Experience-based study | Syllabus proposed by Computing at School | Participant observation  &  semi-structured / unstructured interviews | Exploratory analysis | A good understanding of instructions and conditionals. Problematic approach for loops. |
| [20] | USA | 28 | 4-6 | Informal | One week /half-day (3 hours sessions) coding summer program | Naturalistic observation | Two tablet-based apps: Daisy the Dinosaur and Kodable | Pre- and post-assessments | Quantitative and qualitative data | Foundational coding skills via structured app play |
| [21] | USA | Study1:  58 parents/children dyads  Study2:  Three dyads | 5-7 | Informal | Study 1:  ScratchJr Family Days: single-day events  Study2:  same protocol as study1 | Two case studies: young children & families | Creative coding activities using ScratchJr | ScratchJr projects | Pre-and post-surveys | Algorithms, design process, debugging |

\* N/A: Non-available data

# Appendix C. Studies presentation

ScratchJr studies

Flannery, Silverman, Kazakoff, Bers, Bontá, & Resnick (2013) made their study during the ScratchJr development cycle as the app launched on July 30, 2014, for the Apple iPad. The study had two phases, with 125 children aged 5-7 years took part in informal and formal settings. In the first study phase, the researchers used an app prototype at two pilot study sites. In the baseline study, eighteen of the children learned ScratchJr over three to four half-hour sessions. A second pilot study, with five children, run as a summer enrichment program to assess a new ten-hour curriculum. The results in Phase 1 revealed that many children made more progress in working with ScratchJr than Scratch in terms of meaningful tinkering, software knowledge, and the number of projects they created. The summer ScratchJr sessions revealed that a curriculum based on ‘ScratchJr Cards’ is developmentally appropriate and can help children learn basic coding concepts. Phase 2 of the project had four kindergarten classes and a combined first/second-grade classroom with one hundred children. The participants completed at least nine sessions, with a duration of 30 to 60 minutes per session, depending on each classroom's needs. According to the researchers, ‘the children enthusiastically explored, programmed, and created animated scenes with the ScratchJr. (p.5)’

Strawhacker, Lee, & Bers (2018), in 2014, in a mixed-methods study, explored educators’ attitudes and practices who implemented ScratchJr in their classrooms. Quantitative and qualitative data were collected from all participants. The study involved six teachers and 222 Kindergarten through second-grade students at six schools across the United States. Participants attend two to seven 45-minute sessions of ScratchJr over two months (November–December 2014). The administration of ‘ScratchJr Solve It assessments’ showed that all children showed mid-to-high levels of ScratchJr programming comprehension. According to the researchers, ‘the children successfully attained foundational ScratchJr programming comprehension, demonstrating computational thinking skills of symbol decoding and sequencing comprehension and computational thinking practices of debugging and goal-oriented programming’ (p.361). The researchers also observed statistically significant differences among children in different grades. As expected, older children outperformed younger ones in the tasks. Regardless of their teaching style, the educators reported that the intervention was successful, added that the open-ended software format would allow them to use the ScratchJr in other educational contexts and environments. The teachers also reported that, in general, their students were also enthusiastic about designing their projects in ScratchJr.

Kazakoff (2015) used the ScratchJr to teach traditional literacy practices and digital literacy skills within the standard classroom curriculum, enabling teachers and researchers. The study focused on two kindergartens at a public and a combined first/second-grade classroom at a university's laboratory school. The researcher does not mention the exact date of the study. 18-20 children in each classroom were led through the software for eight lessons up to 2 months. For their 'final project' session, children had 45–60 min to create their project. At the end of the study, the researcher concluded that story creation with the software helped the students develop basic digital literacy skills in developmentally appropriate ways within themes relevant to traditional early childhood curricular. The researcher also stated that an educator could use ScratchJr as a potential tool for introducing digital literacy in the classroom, ‘using a traditional literacy lesson to integrate the technology tool into the general early childhood curriculum while also addressing digital literacy skills.’ (p. 57).

Portelance, Strawhacker & Bers (2016) used the ScratchJr iPad app to teach students basic coding concepts and create their projects. The study participants were sixty-two students in four different classes from kindergarten through second grade at a public elementary school in the US. All participants were exposed to a 6-week ‘Animated Genres’ ScratchJr curriculum, on average twice a week for 1 hour. The paper does not mention the study's exact date. During the study, the students created 977 projects. The researchers analyzed all projects for patterns and differences across grades. The researchers wanted to understand which ScratchJr blocks (commands) the participants used in their projects after completing the curriculum. After analyzing the data, the researchers found that although there were differences between the grades, all children used mostly the ‘Motion’ blocks. Results showed that second-grade students used significantly fewer ‘End’ and ‘Trigger’ blocks than kindergartners. The study also concluded that there were opportunities ‘for further research on ScratchJr's role as an instructional tool in a classroom engaged in learning CT (p.503).’

In this article, Falloon (2016) reports findings from a study exploring the use of ScratchJr in a school numeracy topic (geometry) about learning basic shapes. Children who were working in pairs used ScratchJr block commands to make a character for drawing various shapes and letters. The research involved two primaries (K–6) year 1 and 2 classrooms (5 and 6-year-olds) in a New Zealand primary school from February 2015 to April 2015. Thirty-two students attended five sessions of 25-40 minutes duration. The researcher recognizes that the ScratchJr software, with its open-ended approach and flexible design, may encourage students to construct knowledge through discussion, exploration, and experimentation actively. Falloon recognizing the constraints of the software suggests that as coding activities are likely to be integrated into the curriculum of primary schools, the minor addition of a ‘sprite pencil’ would be helpful as she found that ‘particular features of ScratchJr appeared to influence the extent to which some of these students were prepared to test their ideas (p.589).’

Portelance & Bers (2015) designed and taught an adapted version of the ScratchJr ‘Animated Genres’ curriculum. The participants were sixty-six students in three-second grade classrooms at a public elementary school in the US. 13-daily lessons with one-hour duration engaged students in various skill-enhancing activities designed to help them to build their projects at the end of the study. During the study, the students also conducted artifact-based video interviews in pairs as an alternative approach to the established traditional assessment methods used in other studies. The researchers concluded that their findings add to evidence that students can use the ScratchJr for developing their CT skills. Furthermore, they mentioned that the peer artifact-based video interviews could be used as an alternative assessment method as the students in this study demonstrated a broad range of CT concepts as they talked about their personally meaningful ScratchJr projects.

Thuzar & Nay (2015), for a week in the summer of 2015, had K-2 graders working in half day's activities with the ScratchJr that involved modeling, project planning and creation, and social interaction. The children created a final project as an assessment at the end of the week. ScratchJr was a very flexible tool with relatively 'low floors' and 'wide walls' in the researcher's opinion. Thuzar and Nay also stated that the software was ideal for younger students to get introduced or apply mathematical concepts such as positive and negative integers and reasoning to practical problems. The researchers noted that some students felt restricted by the number of messages sent and the only four pages they could use in their projects. Other students with programming experience in 'drag and drop' based environments mentioned the lack of variables. Another negative comment was the increasing delays that some students faced while trying to insert many sprites on the scene for their project's needs. In conclusion, the researchers commented that ScratchJr could help younger students explore and learn about critical thinking, communication, problem-solving skills, and sharing.

Sullivan, Bers & Pugnali (2017), in their pilot study, used a mixed-method approach to explore young children learning experiences with two different programming interfaces: a graphical coding application (ScratchJr) and a tangible programmable robotics kit (KIBO). The paper does not mention the study's exact date. The researchers wanted to understand whether providing these two different interfaces (graphical and tangible) to young children would affect their CT performance and behaviors and interactions. Twenty-eight children ages four to seven (M=5.86) equally distributed in the KIBO robotics, and the ScratchJr group participated in this study. For five days (approximately three hours daily), children explored the same computational thinking concepts for both groups. At the end of the study, children's understanding of CT skills. Upon completing the week, the researchers measured children's understanding of computational thinking skills (sequencing, repeats, conditionals, and debugging). The study results suggested that children successfully mastered the CT skills taught to both groups, although the user interface type impacts children's learning. However, it is only one of the factors that affect academic and socio-emotional outcomes for children. The researchers concluded that given a cognitively challenging and developmentally appropriate curriculum, both technologies (KIBO and ScratchJr) could foster a positive learning environment helping young children develop their cognitive and coding skills by providing a stimulating environment. Regarding ScratchJr, the researchers mentioned that it enabled children to learn necessary CT skills (such as sequencing, repeating, conditionals, and debugging) and positively affected children's socio-emotional development behavior and attitudes.

Papadakis, Kalogiannakis, & Zaranis (2016) presented small-scale pilot study results, including the use of ScratchJr as a learning tool for teaching CT concepts and necessary coding skills for young children. Forty-three preschoolers (22 boys, 21 girls) in public and a private kindergarten in Greece during the school year 2014–2015 attended a 13hour teaching intervention (twice weekly, one hour per session). The intervention followed the ScratchJr ‘Animated Genres’ curriculum (Portelance & Bers, 2015) and the activities described in ‘The Official ScratchJr Book’ (Bers & Resnick, 2015). Of the 13 hours of course attendance, the first 11 hours used one-to-one tutoring by the researchers, and the other two hours, the children created their projects. The qualitative and quantitative analysis of the data showed that children learned basic CT concepts through using the app. The researchers also found that the app promoted children's collaboration, active learning, and critical thinking skills. Based on these findings, they further recommended the implementation of ScratchJr in earlier education programs but in a developmentally appropriate way: ‘integrating other disciplines, helping children develop cognitive, conceptual, language, and collaborative skills (p.199).’

Kalogiannakis, Ampartzaki, Papadakis & Skaraki (2018) adopted an experimental teaching intervention to examine whether involving hands-on learning activities and creating digital scenarios in the form of ScratchJr projects affects the understanding of gravity and planets children among young children. Seventeen children during a summer school in Greece between July and August 2017 attended a two-weekly teaching intervention with one-hour daily activities. Children created projects in ScratchJr to express their ideas about gravity and planets. The researchers considered ScratchJr as a developmentally appropriate educational tool as it cultivated children's innate interest in learning and introduced them to essential CT and coding concepts while allowing them to express their knowledge of the planets and gravity.

Govind, Relkin & Bers (2020) described two case studies between October 2017 and July 2018 that focused on how young children (ages 5-7) and families recruited to engage in creative activities while using the ScratchJr app. In the first study, fifty-eight parents attended informal coding workshops called ‘ScratchJr Family Days’ with their children and on pre-and post-surveys reported their experiences on the extent to which children take part in the study helped them acquire the CT skills and concepts that ScratchJr promotes: algorithms, design process, and debugging. In the second follow-up study, three families provided qualitative examples of how parents-children enacted their roles and how children exhibited various CT skills in the context of working collaboratively. Studies suggest that when children are actively involved in ScratchJr activities, they can express their creativity and expand their CT concepts knowledge. These findings also promoted ScratchJr as a playful, developmentally appropriate platform for young children to learn to code and engage in foundational CT skills (Govind, Relkin & Bers, 2020).

Sheehan, Pila, Lauricella & Wartella (2019) observed 31 parents and their 31 4.5- to 5.0-year-old children, exploring how they collaboratively use coding apps for learning and whether specific parent-child interactions in families promote children's learning from screen-based apps. The paper does not mention the study's exact date. The participants used the PBS KIDS ScratchJr and coded for spatial talk, question-asking, task-relevant talk, and responsiveness. Results showed that coding apps may be a rich context for STEM learning and that specific parent-child interactions play a significant role in children's learning progressions while interacting with STEM apps on touchscreen devices. The research also added to growing evidence that young children can learn coding skills from apps like ScratchJr.

Strawhacker & Bers (2019), in a mixed quantitative and qualitative case study, examined the cognitive effects of introducing coding activities to young children. Fifty-seven children in kindergarten through the second grade of a public school in the US engaged in 12-hour lessons, twice-weekly, for six-week lesson activities adapted from the Animated Genres curriculum Group (Portelance et al., 2015). The paper did not mention the study's exact date. A post-intervention assessment assessed children's programming comprehension and knowledge using measures of ScratchJr. Results showed that all students mastered foundational coding concepts, although there were marked differences in performance and comprehension across the different grades. Younger children may be developmentally unable to master complex ScratchJr concepts such as flow blocks to the same degree as older children. Among other findings, the researchers found that the children were deeply engaged in problem-solving and social reasoning while working on their ScratchJr projects.

Chou (2020) adopted a quasi-experimental pretest and posttest control group study with qualitative observation support to determine the effectiveness of instructional approaches with ScratchJr for elementary school students in Taiwan. Twelve third graders for eight weeks completed eight projects in class and take-home written assignments. The paper did not mention the study's exact date. A simplified version of Brennan and Resnick's (2012) framework was used to document young children's programming learning as a posttest and as an extra measurement four weeks after the last training session. The quantitative results showed that the students significantly improved their computational literacy. The participants kept much of the knowledge acquired from the study one month after its completion. The qualitative data suggested that the ScratchJr interface helped students instill necessary skills, such as coding skills, innovation, and creativity. Some students expressed displeasure about the limited range of ScratchJr features.

Lowe & Brophy (2019) study involved eighteen participants of one first grade class in a Title 1 school located in the United States. The class engaged in two half-day computational thinking lessons. The first half-day lesson included seven activities, exposing participants to creating sequential only algorithms using a programmable hardware device. The second day expanded to introduce ScratchJr, allowing for open-ended storytelling animations. The paper does not mention the study's exact date. Each student had an iPad to retell a story in ScratchJr independently. Using qualitative analysis, the researchers evaluated how children's stories change considering learning how ScratchJr enables animation and considers the role of programming in learning CT. The researchers concluded that students seemed to genuinely enjoy using ScratchJr for storytelling as they were focused and worked the entire time. They also commented that this is not always the case with demanding tasks and the specific age group. The participants' final work varied in complexity and quality, although it seems clear that growing knowledge of the capabilities of ScratchJr modified their use of the medium. Study results showed that ScratchJr is a powerful accelerator for understanding computation than abstract storytelling alone and that ScratchJr provides a powerful tool for an open-ended play that relates to both programming and literacy if used correctly.

Herheim & Severina (2020) involved first-year pre-service teachers and year four students focused on links between ScratchJr functions and students' mathematical explanations and justifications. The intervention consisted of two mathematics lessons, and the students worked in pairs and shared a tablet (the paper does not mention the study's exact date). The student's task was to use multiplication tables (corresponding number sequences) to navigate characters when creating an animation in ScratchJr. According to the researchers, ScratchJr mediates explanations and justifications by making several representations of mathematical ideas simultaneously available through different modalities. The study results showed that despite the constraints of ScratchJr, the software has some functions that can foster mathematical argumentation. Nevertheless, they also mention that the students' engagement in mathematical explanations and justifications requires teacher awareness about supporting the students’ work and appropriate mathematical tasks.

Kodable study

In Turkey, Karadeniz, Samur, and Özden (2014) examined whether five years children can learn the basics of programming (sequence, conditions, and loops) by playing with algorithms with the Kodable app. This case study was done with 25 five-year-old kindergarten children at two separate classes for three weeks (the paper does not mention the study's exact date). Specific lesson plans combined traditional teaching techniques with games in the classroom and the coding app Kodable. Findings demonstrated that the students enjoyed playing with Kodable, and it was also easy for them to develop basic algorithms through the game using more sophisticated programming concepts such as the if-then-else statement. However, they had difficulty in the loop statement. The researchers suggested using both digital and in-class games to foster algorithmic thinking for kindergarten children.

Daisy the Dinosaur and Kodable study

Pila, Aladé, Sheehan, Lauricella, & Wartella (2019), in their study, looked at children's ability to learn necessary coding skills (sequencing, conditions, and loops) using two commercially available apps, Daisy the Dinosaur and Kodable. Twenty-eight children (4- to 6-years old) enrolled in a half-day (three hours sessions) coding summer program in 2016 in the US. The test scores of children's in command knowledge and gameplay from pretest to posttest revealed that, on average, participants significantly improved their knowledge of Daisy commands. Similarly, participants showed significantly higher scores in progression and Kodable gameplay after the course. The researchers concluded that these two apps could be a valuable tool to teach young children critical coding skills while providing an attractive environment conducive to enjoyable learning for both genders.

Lightbot and ScratchJr study

Rose, Habgood & Jay (2017) conducted an exploratory study exploring young children's programming approaches using two coding tools with contrasting interfaces, ScratchJr and Lightbot. To consider the impact of programming approaches on children's CT skills, the researchers created two versions of a Lightbot style game for six and 7-year-olds children. One version used the Lightbot programming interface, and the other the ScratchJr-like interface. The participants were 20 boys and 20 girls between 6 years from a large primary school in a low-income area in northern England. The paper does not mention the study's exact date. The results showed that the student’s overall performance was similar in both versions of the game. All students in both games reached a similar level, spent a similar amount of time on each level, and took a similar number of attempts to complete each level. The researchers found indications that children aged six and seven interacted differently with the ScratchJr-like programming interface compared to the Lightbot interface. Children using the ScratchJr-like interface performed more program manipulation, which according to the researchers, could show a ‘more bottom-up or bricolage, programming approach’ (p.306) in line with the constructionist foundations of Scratch and ScratchJr. A more surprising finding was that more able children (as measured by non-verbal reasoning skills or game performance) performed more tinkering than their low ability counterparts, suggesting that ‘they were more suited to the free-design approach of ScratchJr-like instructions (p.306).’ This difference was only found in players using the ScratchJr-like interface and not the Lightbot players.

Lightbot and Kodable study

Gomes, Falcão, & Tedesco (2018) in Brazil used six coding apps to examine how they help young children to understand programming concepts. Forty-two students aged 5 to 7 years attended nine weekly 40- minute meetings. The paper does not mention the study's exact date. Data were collected through participant observation and semi-structured and unstructured interviews, focusing on the children's main difficulties interacting with the apps in classroom settings. The researchers found that the approach adopted by Lightbot for the loops was one of the most cognitive demanding for children. Lightbot, differently from the other coding apps, does not provide a specific command for the repetition concept. Indeed, instructions must insert in a procedure that must have a recursive call until all tiles are highlighted. None of the participants could accurately describe how these commands functioned or correctly use the interface elements related to repetition. Kodable repetition approach was also tricky for the children to understand as they needed to 'insert instructions one below the other and use the plus (+) or minus (-) signs to indicate the number of repetitions' (p.88)’.

Similarly, none of the participants managed to use the command correctly, complaining that they did not understand how to use it. On the contrary, comments analysis showed that the students demonstrated a satisfactory comprehension of the conditional concept in Kodable as they were able to describe the functioning of the interaction element. Regarding Lightbot, the researchers remarked on the complexity of language (both visual and textual) for the age range. They mentioned that the 'complex constructs in initial tutorials, help, and feedback' were encountered in Lightbot (p.88).

Daisy the Dinosaur and ScratchJr study

Falloon, Hale, & Fenemor (2016) study combined the use of the Daisy the Dinosaur and the ScratchJr app to explore how teachers plan and integrate coding into their numeracy program (geometry) for students to learn basic shapes. The study took place from February to September 2015, and the participants were 36 young students from a primary school in New Zealand. Findings suggest using Daisy the Dinosaur app for learning the ‘mechanics’ of building code. The more structured and less optioned environment of Daisy provides very few instructions to students than ScratchJr, helping students tackle the basic concepts they needed. The ScratchJr open-ended approach problematizes students, not enabling them to create code to handle the tasks with relative ease. All students seamlessly applied the techniques learned using Daisy to develop code to create basic geometric and letter shapes in ScratchJr. The researchers concluded that while ScratchJr had some evidence for effectiveness when used to help students learn about shapes, it needs to be supplemented by other concrete materials in the classroom as students found it challenging to visualize the shape and dimensions of objects their code created.