Leveraging Web Components for Authoring Interactive Mathematics

Rudolf Hoffmann and Edgar Seemann
Furtwangen University, Germany

Keywords: Mathematics, Interactivity, Authoring, e-Learning, Web Components.

Abstract: Digital, interactive content can support active learning and provide both motivation and an automated feedback to students. Unfortunately authoring interactive content remains a difficult task for teachers. There are few interoperable standards and e-learning platforms often restrict what is technically possible. Even if some teachers create amazing interactive content it is challenging to publish and share this content with colleagues. This paper proposes a way to leverage *Web Components*, a rather new technology supported by modern browsers, to allow teachers to quickly author interactive exercises involving mathematical expressions and visualizations. As *Web Components* are standardized they can be shared and embedded in any website. The proposed *Web Components* for mathematics allow for: 1. A convenient visual input method for expressions and formulas, 2. A sophisticated validation system for these expressions in order to give immediate feedback to students on their solutions.

1 INTRODUCTION

The field of mathematics remains one of the most challenging areas for e-learning and e-teaching. While there is a body of products and research on teaching tools (MapleSoft, 2014; Hastings et al., 2015), content creation (Leathrum, 2010; Seemann, 2016; Melis et al., 2009), math typesetting (Ausbrooks et al., 2014; Knuth, 1986; Kohlhase, 2003), assessment tools (Perfect, 2015; Sandene et al., 2005; Seemann, 2014; Joglar et al., 2013) and effects on learning outcomes (Beal et al., 1998; Aleven et al., 2006), the field still lacks proper interoperable standards. This makes content creation and sharing cumbersome for teachers.

1.1 Content Creation

Firstly, it is a complex task to author interactive math content. E-Learning platforms such as Moodle, OpenOLAT etc. (Moodle, 2022; OpenOLAT, 2022) try to support teachers and offer possibilities to create interactive exercises and tests. The types of interactive exercises which can be implemented this way, however, is very limited. The main functionality of the leading e-learning platforms still seems to be mostly centered around multiple choice questions or questions with fixed, unambiguous answers.

While displaying formulas and equations has gotten easier with modern libraries such as MathJaX or KaTeX (MathJAX, 2022; KaTeX, 2022), the problem of having students enter math expressions with fractions, powers, roots etc. is in many cases impossible or at least very cumbersome in these systems. There is also only very limited support for validating math expressions with all their equivalent representations. Plugins such as STACK (STACK, 2021) try to remedy this situation somewhat. The capabilities are, however, still far from what teachers would like to have.

Commercial publishers of e-learning content and electronic books typically create interactive material based on their own proprietary technologies. The results can be quite impressive. When teachers use this content, they are often limited to a single provider whose license the school acquired and it is mostly impossible to adapt, modify, build upon or improve the interactive content in meaningful ways.

1.2 Content Sharing

The limited availability of free interactive content can also be attributed to the difficulty of sharing and exchanging content with colleagues. Often the content is locked to a specific commercial or open-source platform, which others may not have access to and therefore cannot use. The QTI standard (1EdTech, 2016) allows sharing of simple exercises between e-learning platforms via an XML based exchange for-

Hoffmann, R. and Seemann, E.
Leveraging Web Components for Authoring Interactive Mathematics.
DOI: 10.5220/0011946600003470
In *Proceedings of the 15th International Conference on Computer Supported Education (CSEDU 2023) - Volume 1*, pages 264-271
ISBN: 978-989-758-641-5; ISSN: 2184-5026

mat. The author has to export the interactive created content from his/her e-learning platform as a ZIP archive containing the XML-based description. Other teachers can then import this content in their e-learning system given that both support the same QTI standard and are sufficiently compatible. In practice, this process is sufficiently difficult for many teachers to avoid going through this export and import process and simply not share content.

1.3 Authoring for the Web

An alternative approach for interactive content is to author content directly for the web instead of using e-learning platforms. While this typically requires content authors to have a deeper technical knowledge, it offers much more flexibility. We can create a much wider variety of types of experiences and above all, it is easy to publicly publish web content and share it with others.

In this paper we propose to leverage the power of web components to simplify the process of content creation and sharing for authors. Web components allow us to extend the set of available HTML tags. We can define math specific components which support content creation with math specific functionalities. This allows authors to quickly and easily create web-based interactive math experiences that can be used and shared in any web site.

The goal is to show how authors can be provided with web components which automatically integrate the following interactive functionalities:

1. A visual input method, which allows the entry of math expressions e.g. fractions, powers, roots, vectors etc.
2. A way to integrate interactive input fields at any position in a mathematic expression.
3. A sophisticated validation system, which allows authors to define how user input should be tested for correctness and to provide feedback to students

To accomplish these goal we combine the technology of web components with a specialized math library written in JavaScript. This math library provides authors with pre-built authoring functionalities.

2 WHAT ARE WEB COMPONENTS?

Web components are a technology which allows developers to extend HTML with new, reusable and sharable HTML-Tags. These tags can provide not only a custom look, but also component specific functionality. Before we dig deeper into this technology, let us briefly revisit how web pages and HTML tags work.

Every web page consists of a set of (possibly nested) HTML tags, which define the structure and, at least to some extent, the appearance of the page. For example:

```
<!doctype html>
<html>
  <head>
    <title>Math Test</title>
  </head>
  <body>
    <h1>An example HTML page</h1>
    <label>How many corners has a
           pentagon?</label>
    <input name="answer">
  </body>
</html>
```

There are e.g. tags for the headings on the page (`<h1>`,`<h2>` etc.) and for user input (`<input>`). Authors compose their content based on these basic tags, which are understood by every browser. One can find a list of all standardized HTML tags at `https://www.w3schools.com/TAGS`. In order to add interactivity or go beyond what is included within these HTML tags, authors need to become programmers and learn JavaScript, the standard programming language of the web. As a consequence, until now it was impossible to create truly interactive math content for the web without being a programmer or resorting to a specialized e-learning platform/authoring tool.

In this paper we propose custom web components for authoring interactive maths, which allow every teacher who is able to write basic HTML to become an author for interactive web content. Our math web components provide additional HTML tags such as `<mwc-expr>` and `<mwc-viz>`, which augment the browser's default capabilities.

Imagine the example of a teacher wanting to display a simple algebraic math exercise with fractions. By using the newly available web components, he/she can do so by writing the following short HTML snippet with a LaTeX expression:

```
<mwc-expr>
 display: "\frac{1}{2}+\frac{2}{3}=\ph{}",
 solution: "\frac{7}{6}"
</mwc-expr>
```

The web component for the tag `<mwc-expr>` will automatically take care of parsing and rendering ex-

pression correctly (see Figure 1). The `\frac` statements will be rendered as a fraction and the `\ph` statement, which stands for placeholder, will be rendered as an interactive input field. These input fields will be automatically validated as their input changes. This means, as soon as a student enters an answer he or she will get an immediate feedback on his/her response. This can be in the form of a coloring (red for incorrect and green for correct) as well as text feedback. In order to allow students to easily enter math expressions, the web component also displays a virtual keyboard at the bottom of the page. This virtual keyboard contains special keys for fractions, powers, roots, vectors etc.

Figure 1: A basic math exercise with fractions and an automatically validated input field.

Note that, while authors can write the HTML tags directly there is also a graphical editor for the proposed web components, which allows for a more convenient authoring experience. In the editor expressions can be entered visually by the authors via the virtual keyboard. Additionally, authors can specify solutions for every placeholder alongside a method of validation e.g. how many decimal places are required or whether a fraction needs to be canceled. This authoring tool then allows to save the resulting content as a simple HTML page with the included custom web components.

In this paper, we will mostly focus on the underlying technology and ideas of the proposed web components. We see this work as just the beginning and we hope that other e-learning researchers and developers will adopt a similar approach to provide teachers with a wide range of web components to create rich, interactive learning experiences.

2.1 Implementation of Web Components

Web components have been getting more and more popular recently, with many web component libraries such as Shoelace Components (Shoelace, 2022)

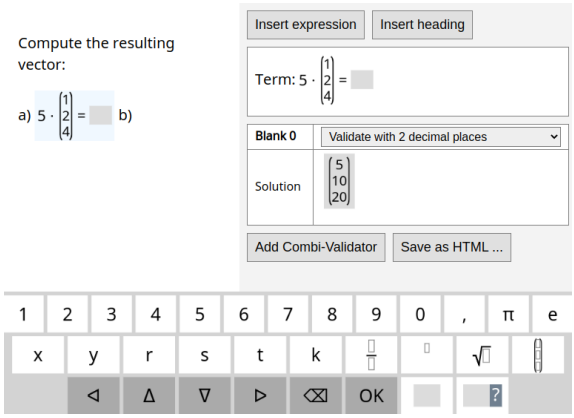

Figure 2: The graphical user interface allowing authors to create interactive pages without the knowledge of HTML.

and Material Web Components (MaterialWeb, 2022) readily available for web content authors. These existing component libraries focus on general web content and our goal here is to leverage this technology specifically for interactive math content.

In order to implement our custom web component, we need to extend a basic HTML-Tag with math-specific functionality. This is accomplished by writing JavaScript code which inherits from the base class ”HTMLElement”:

```
class MathComponent extends HTMLElement {
  constructor() {
    super()
    ...
  }
}
```

Additionally, we need to register this new class as a custom element within the browser:

```
customElements
   .define("mwc-expr", MathComponent)
```

In the above example `<mwc-expr>` will be the name of the tag associated with the component. Note that, tag names must contain a dash to avoid collisions with already existing tags in the browser.

The constructor of the component is called when the element is created. This happens e.g. when the browser encounters the new tag `<mwc-expr>` in a HTML page.

Let us look again at our example from section 2:

```
<mwc-expr>
 display: "\frac{1}{2}+\frac{2}{3}=\ph{}",
 solution: "\frac{7}{6}"
</mwc-expr>
```

In this case the constructor must first parse what is enclosed within the `<mwc-expr>` tag. This can be accomplished by reading the element's `innerText`

property. Note that, the inner text in our case is actually in a JSON format and can be parsed with the the browser's `JSON.parse` function once we add curly braces:

```
constructor() {
  super()
  this.data =
     JSON.parse(`{${this.innerText}}`)
  ...
  this.appendChild(renderedExpression)
  this.onclick = ev => { ... }
  this.onkeydown = ev => { ... }
}
```

Once parsed, the `this.data` variable contains all the information necessary to render the component. In the next steps we parse the LaTeX expressions in `data.display` and `data.solution`. The "display" expression is rendered and appended to the element by calling the `appendChild` function of the base class `HTMLElement`. Finally, we register event listeners for mouse clicks and keyboard events. This allows the component to process user input and update the component accordingly. That is e.g. filling in the response that is typed as well as validating the solution automatically on every key press.

Figure 3: A basic math exercise with fractions and an automatically validated input field.

Note that, LaTeX does of course not have a `\ph` command. This has to be considered both when parsing and rendering an expression. Also note, that we deliberately did not discuss the usage of a shadow DOM, as this is a technical detail, which is of minor importance for our discussion on the advantages of web components for authoring interactive exercises.

Finally, in order to to make a new web component available on a web page we need to load its JavaScript code in a `<script>` tag. We can, of course, bundle multiple web components into a single .js file and load them in a single step:

```
<script src="mwc.js"></script>
```

Now that we have some basic technical understanding of what web components do and how they work, let us focus back on what makes interactive maths particularly challenging. And let us see how the proposed web components can help authors in creating interactive content. We consider mainly three areas:

1. Math input: How to allow students to enter more complex mathematical expressions.
2. Math validation: How to enable authors to specify how mathematical expressions should be validated.
3. Math visualizations: How can authors create interactive math visualizations e.g. for function graphs.

3 MATH INPUT

Math-based exercises often require students to compute values, expressions or functions, which subsequently need to be entered into an input field. Unfortunately, there is no common standard on math input. The default HTML input fields only support simple text. While it is possible to create many expressions as plain text using operators such as ^,*,+ etc., this is neither intuitive nor is it clear what syntax to use for roots, vectors, integrals etc.

Web components can implement their own way of user input by reacting to browser events for the mouse and keyboard. Moreover, as shown in the figures of this paper previously, they can also implement a virtual keyboard, which allows students to easily enter more complicated math expressions such as fractions, powers or roots with special keys on the virtual keyboard. These expressions are then rendered visually without requiring students to learn some special syntax. When pressing the fraction key on the virtual keyboard e.g. a fraction with empty boxes for numerator and denominator appears, which can be subsequently filled with the appropriate values either using the virtual keyboard or a physical keyboard:

A virtual keyboard can provide special keys for any kind of mathematical construct. Other common examples are powers, roots and vectors. The virtual keyboard may even adapt to the current exercise. E.g. if the solution does not contain any powers, roots or vectors an implementation of the web component may choose not to display those keys on the virtual keyboard.

Virtual keyboards are a well known concept on mobile devices where this type of input will feel very natural. On more traditional laptop and desktop computers the virtual keyboard acts as an extension of the

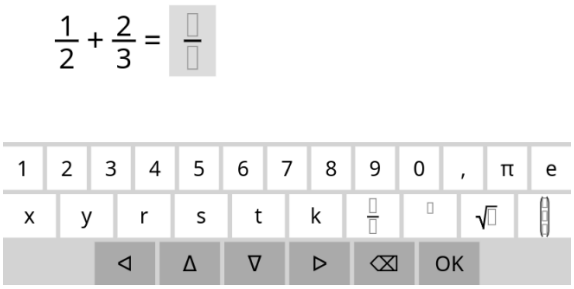

Figure 4: Entering a fraction by pressing the fraction key on the virtual keyboard will produce a visual fraction display with empty boxes for the numerator and denominator.

physical keyboard, which of course does not have special keys for more sophisticated math expressions.

In summary, instead of being restricted to what a standard HTML `<input>` element can do, a custom web component can support a visual input method, which allows an intuitive way of data entry in interactive exercises.

4 MATH VALIDATION

Interactive exercises should be able to automatically react to student responses and provide feedback on their solution attempts. The first step in this process is to allow students to easily and intuitively enter math expressions (see section 3). The next step is then to analyze the student's response and compare it with the solution of the exercise.

At first glance, this may seem to be a straight forward and obvious task. But representations in mathematics are not unique and a feedback system should understand any equivalent representation of the solution. This is true for numerical values such as 0.5 vs $\frac{1}{2}$ vs. $\sqrt{\frac{1}{4}}$, but even more for math functions such as polynoms e.g. x^2+2x+4 vs $4+2x+x\cdot x$, exponentials etc.

Depending on the circumstances authors may want equivalent representations to be accepted as as correct or incorrect. Let us e.g. consider the simple exercise $\frac{1}{6}+\frac{3}{4}$. The solution is $\frac{11}{12}$, but many students might rather compute the numerically equivalent expanded version $\frac{22}{24}$. An author may frame the associated question in a way that requires students to fully cancel their solution or allow expanded variants depending on what educational goal he or she has in mind. The same is true as to allowing decimals such as 0.5 instead of fractions.

A validation system therefore has to give an author the flexibility to specify not only what the solution is, but also what representations of the solution should be accepted. For our web components we therefore introduced the concept of so-called *validators*. A *validator* is a JavaScript method which takes the solution and a student response and computes, whether the response is correct. Optionally it outputs a text-based feedback or hint. For example:

```
function canceledFrac(solution, userInput)
{
  if (solution.value!=userInput.value)
    return { valid: false }
  else if
  (solution.numerator!=userInput.numerator)
    return {
      valid: false,
      msg: "Fully cancel the fraction" }
  ...
  else return { valid: true }
}
```

The above example validates a fraction and only accepts the solution if is fully canceled. If it has the correct value, but seems to be an expanded representation the validator returns a feedback message for the student on what to do.

We bundle the proposed web components with a math library containing a set of validators for authors to use. These validators can be either specified in the HTML tag or in the graphical editor (see Figure 2). In order to use the above validator, we write:

```
<mwc-expr>
 display: "\frac{1}{2}+\frac{2}{3}=\ph{}",
 solution: "\frac{7}{6}",
 validator: mwc.validators.canceledFrac
</mwc-expr>
```

If no validator is specified a matching default validator is chosen based on the solution value. Available validators include functionality for rounded numbers, exact numbers with roots e.g. $2+\sqrt{3}$, fractions, polynoms e.g. x^2+2x+3, vectors etc.

4.1 Parameterized Validators

With JavaScript it is also easy to create validators with parameters. Consider, for example, a validator for decimals where the author wants to specify the minimum number of decimal places a value should have. In JavaScript this can be achieved by a so-callled higher-order function, which returns a validator function based on the provided parameter. The programmatic structure looks like this:

```
function decPlaces(count) {
  return function(solution, userInput) {
    roundedSol = solution.toFixed(count)
```

```
    ... // further validation code
  }
}
```

Using this paramterized validator for the author is straightforward. This example would require a student to enter the result as a decimal number with at least 3 decimal places:

```
<mwc-expr>
 display: "\frac{1}{2}+\frac{2}{3}=\ph{}",
 solution: "\frac{7}{6}",
 validator: mwc.validators.decPlaces(3)
</mwc-expr>
```

4.2 Multi-Input and Combi Validators

Math expressions may also contain multiple placeholders for user input. In this case authors provide the web components with a list of solutions and optionally a list of validators. Of course these lists have to be in the same order as the placeholders appear in the expression. For the fraction example used throughout this paper, there would be a separate input field for the numerator and the denominator:

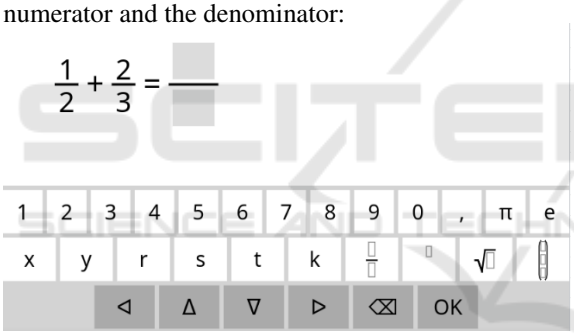

Figure 5: A math exercise with two separate user inputs.

In order to obtain this variant of the exercise we modify the HTML tag in the following manner:

```
<mwc-expr>
  display: "\frac{1}{2}+\frac{2}{3}
            =\frac{\ph{}}{\ph{}}",
  solution: ["7", "6"],
</mwc-expr>
```

Note that, in this case the two input fields act completely independent and we cannot e.g. give feedback on whether the result is OK, but should be canceled. A similar issue arises, when considering the two solutions of a quadratic equation. It shouldn't matter in which order the two solutions are entered by the student. But when the input fields are considered independent each input has its fixed solution, which cannot be switched.

To overcome this issue in our web components, we introduced the concept of *combi validators*. A combi validator is responsible of validating multiple input fields. This way it can consider the dependencies between these input fields. In the case of the quadratic equation it can therefore also detect in which order the two solutions have been provided.

Using a combi validator is as easy as using a normal validator:

```
<mwc-expr>
  display: "\frac{1}{2}+\frac{2}{3}
            =\frac{\ph{}}{\ph{}}",
  solution: ["7", "6"],
  validator: mwc.validators.combiFrac
</mwc-expr>
```

4.3 Custom Validators

It is also possible for authors to create their own validators by implementing a custom JavaScript function. Obviously this requires programming knowledge. An example for a custom exercise specific validator in the web component could look like this:

```
<mwc-expr>
 display: "\frac{1}{2}+\frac{2}{3}=\ph{}",
 solution: "\frac{7}{6}",
 validator: (solution, userInput) =>
 {
    ... // custom validation logic
    return { valid: true }
 }
</mwc-expr>
```

5 MATH VISUALIZATIONS

Next to web components for algebraic exercises, we also propose a component for graphing and geometry. Our interactive graphing component can display a paramterized function graph and allow students to play with sliders to modify the parameters.

An author can use the associated `<mwc-viz>` tag and provide a function term with parameters, as well as intervals for the ranges of these parameters. As before, this information will be parsed by the web component and an interactive graph is created oon the web page automatically. The HTML code could e.g. be:

```
<mwc-viz>
  func: "a*sin(b*x+c)",
  a: "[1,4]",
  b: "[0.1,5]"
  c: "[-2,2]"
</mwc-viz>
```

The resulting component is displayed in the browser as follows with the interactive sliders in the upper right corner of the graph:

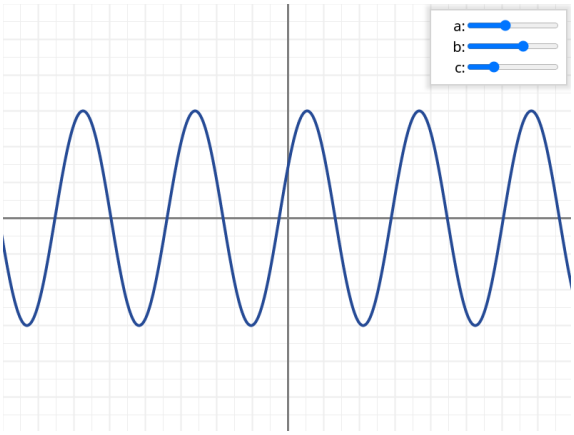

Figure 6: An interactive graphing component.

An alternative to such a web component is certainly the proprietary tool GeoGebra (GeoGebra, 2020), which allows the embedding of graphs in web pages and has a much broader feature set. Web components are, however, freely customizable and teachers or organizations can self-host their web components. Thus, not having issues with external servers, licensing and data protection laws.

The goal of this paper is also to showcase the potential of the technology rather than to claim that the proposed web components are superior to all existing e-learning tools.

6 CONCLUSION

In this paper we propose to use the technology of web components to support teachers in creating interactive math content. While authoring for the web has many advantages as far as interoperability, publication and sharing is concerned, it was so far quite involved and authors mostly had to be programmers to create interactive experiences.

Web components as a technology can make content creation for teachers much easier by providing math specific components which augment what standard HTML is capable of. We have shown different use cases for web components for both algebraic exercises and math visualizations. We hope that more e-learning developers and researchers adopt this technology and provide authors with amazing web components to use.

The proposed web components of this paper are currently used in a publicly funded project to create a large database of randomized interactive math exercises. The acknowledgements for this project will be added after the double-blind review process.

REFERENCES

EdTech (2016). *QTI 2.1 - 1EdTech Question and Test Interoperability*.

Aleven, V., McLaren, B., Sewall, J., and Koedinger, K. (2006). The cognitive tutor authoring tools (ctat): Preliminary evaluation of efficiency gains. In *ITS 2006*.

Ausbrooks, R., Buswell, S., Carlisle, D., and Chavchanidze, G. (2014). Mathematical markup language (mathml) version 3.0 2nd edition. http://www.w3.org/TR/MathML3.

Beal, C., Beck, J., and Woolf, B. (1998). Impact of intelligent computer instruction on girls' math self concept and beliefs in the value of math. In *Annual meeting of the American Educational Research Association*.

GeoGebra (2020). Geogebra - powerful math apps. https://geogebra.org.

Hastings, C., Mischo, K., and Morrison, M. (2015). *Hands-on Start to Wolfram Mathematica and Programming with the Wolfram Language*. Wolfram Media Inc. ISBN: 9781579550776.

Joglar, N., Risco, J. L., Sánchez, R., Colmenar, J. M., and Díaz, A. (2013). *Testing in Math courses a new tool for online exams automatically graded: a teaching and learning experience*.

KaTeX (2022). Katex - the fastest math typesetting library for the web. https://katex.org.

Knuth, D. E. (1986). *Computers & Typesetting, Volume B: TeX: The Program*. Addison-Wesley.

Kohlhase, M. (2003). Toward openmath version 2. In *Mathematics on the semantic web*.

Leathrum, T. (2010). Math authoring for the web made easier. In *Convergence (Mathematical Association of America)*.

MapleSoft (2014). *E-Book: Clickable Calculus Study Guide*.

MaterialWeb (2022). Material web components catalog. https://material-components.github.io/material-web.

MathJAX (2022). Mathjax - beautiful math in the browser. https://mathjax.org.

Melis, E., Goguadze, G., Libbrecht, P., and Ullrich, C. (2009). Activemath - a learning platform with semantic web features. In *Ontologies and Semantic Web for e-Learning*.

Moodle (2022). Moodle learning management system. https://moodle.org.

OpenOLAT (2022). Openolat - online learning and training. https://olat.org.

Perfect, C. (2015). A demonstration of numbas, an e-assessment system for mathematical disciplines. In *Internationl Conference on Compupter Assisted Assessment*.

Sandene, B., Bennett, R., Braswell, J., and Oranje, A. (2005). *Online Assessment in Mathematics*.

Seemann, E. (2014). Teaching mathematics in online courses - an interactive feedback and assessment tool. In *International Conference on Computer Supported Education*.

Seemann, E. (2016). Mathauthor: Authoring interactive math exercises for the web. In *International Conference on Computer Supported Education*.

Shoelace (2022). Shoelace - a forward-thinking library of web components. https://shoelace.style.

STACK (2021). Stack - online assessment. https://stack-assessment.org.