

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Фронт-энд разработка

Отчет по проекту

Выполнил:  
Рейнгеверц В. А.  
К33401

Проверил:  
Добряков Д. И.

Санкт-Петербург

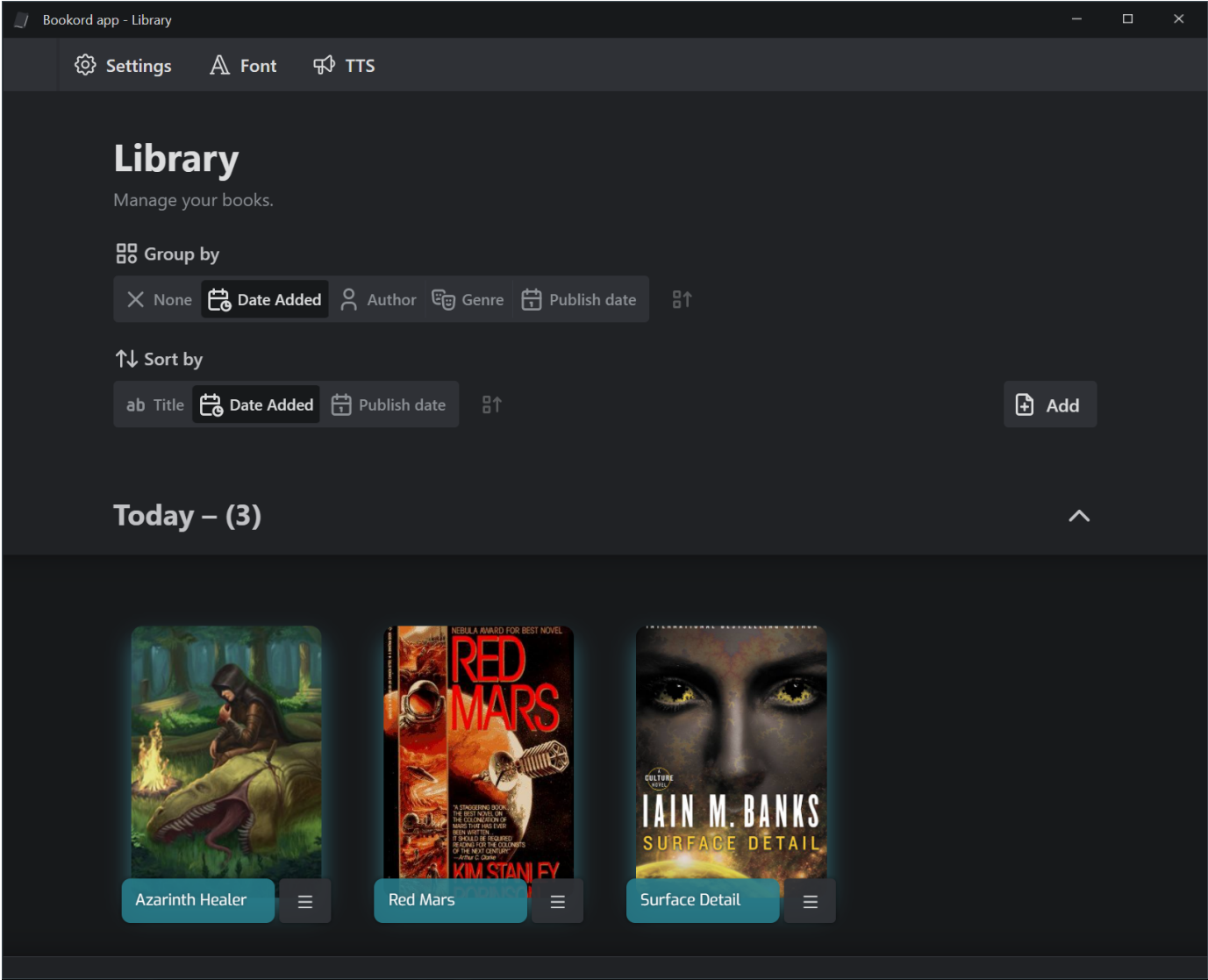
2022 г.

# Отчет по проекту

Рейнгеверц В.А. – K33401

## Предметная область: Bookord

Библиотека



Bookord app - Read

← Settings Font TTS

Red Mars

Chalmers got up to move on, meeting for one instant Selim's insistent gaze. Then he strode down a side street, one of the narrow lanes that connected the city's seven main boulevards. Most were paved with cobblestones or streetgrass, but this one was rough blond concrete. He slowed by a recessed doorway, looked in the window of a closed boot manufactory. His faint reflection appeared in a pair of bulky walker boots.

Selim el-Hayil's reflection appeared among the boots.

"Is it true?" he demanded.

"Is what true?" said Frank crossly.

"Is Boone anti-Arab?"

"What do you think?"

"Was he the one who blocked permission to build the mosque on Phobos?"

"He's a powerful man."

The young Saudi's face twisted. "The most powerful man on Mars, and he only wants more! He wants to be king!" Selim made a fist and struck his other hand. He was slimmer than the other Arabs, weak-chinned, his moustache covering a small mouth. A bit of a rabbit, but with sharp teeth.

"The treaty comes up for renewal soon," Frank said. "And Boone's coalition is bypassing me." He ground his teeth. "I don't know what their plans are, but I'm going to find out tonight. You

Part 1. Festival Night16/3717/806

Bookord app - Settings

←

Settings

Font

TTS

# Settings

Customize how the app feels.

App Settings

Theme

A

Font

Style font's look and size.

☐

Override — Font family

Override the font family of the book.

Comic Sans MS

☐

Override — Line height

Override the amount of vertical space between lines of text in a paragraph.

1.20

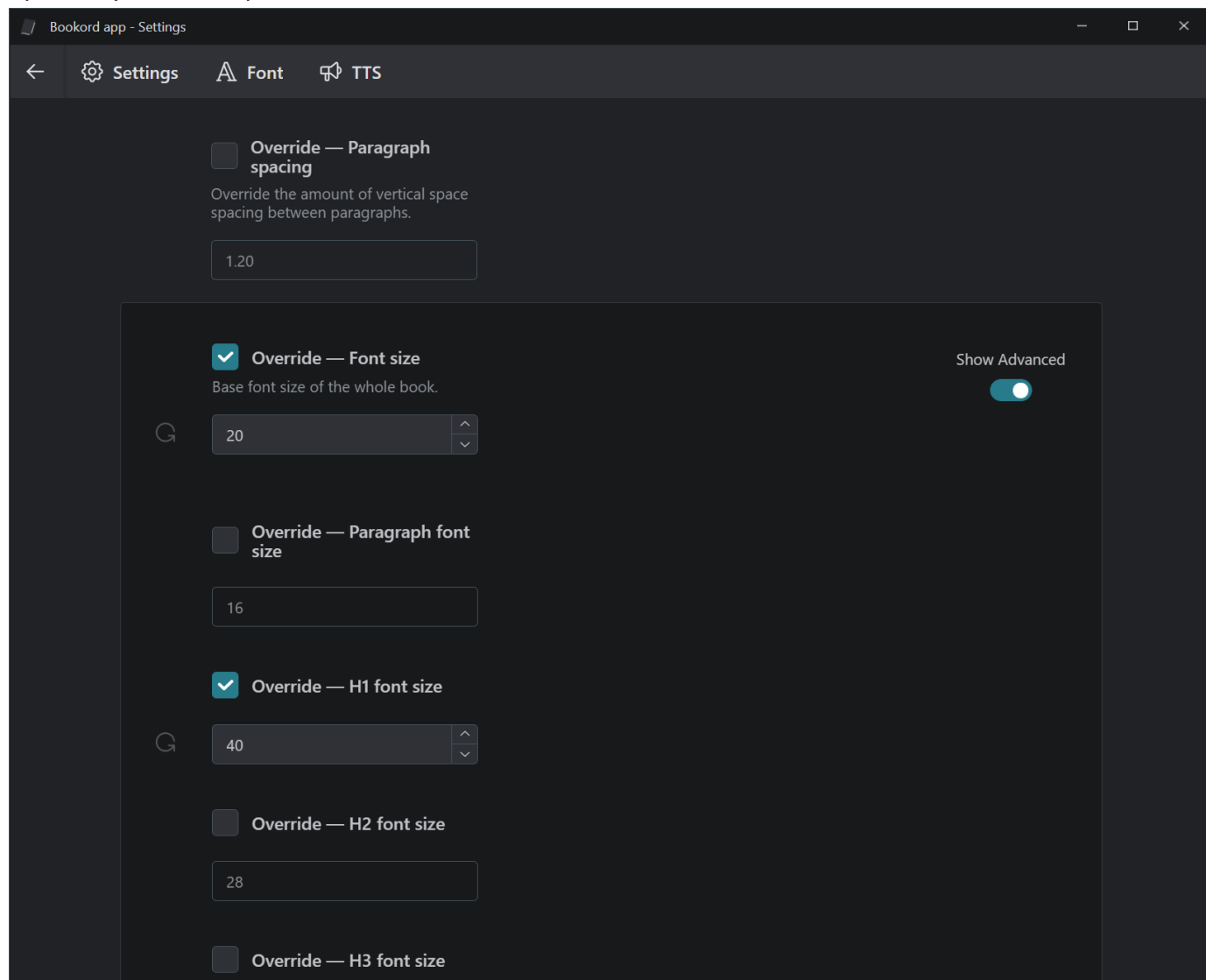
☐

Override — Paragraph spacing

Override the amount of vertical space spacing between paragraphs.

1.20

## Продвинутые настройки



**Bookord** – это многофункциональное приложение для чтения книг с открытым исходным кодом для ПК. У него интуитивный интерфейс и множество полезных функций, позволяющих сделать чтение электронных книг более удобным.

С **Bookord** можно читать любые книги с компьютера, не беспокоясь о том, что можно потерять страницу на которой остановились, и не носить с собой физическую книгу.

С помощью данного приложения можно читать, аннотировать, выделять, организовывать свою библиотеку, пользоваться продвинутыми опциями преобразования текста в речь, персонализировать интерфейс и многое другое!

## Основные библиотеки и фреймворки

Основанно на шаблоне:

[reZach/secure-electron-template](#) | [GitHub](#)

- Electron
- React
  - [Mantine](#) (UI библиотека)
  - [tabler-icons-react](#) (SVG иконки)
- [tinycolor2](#) (для ‘математики’ с цветами)

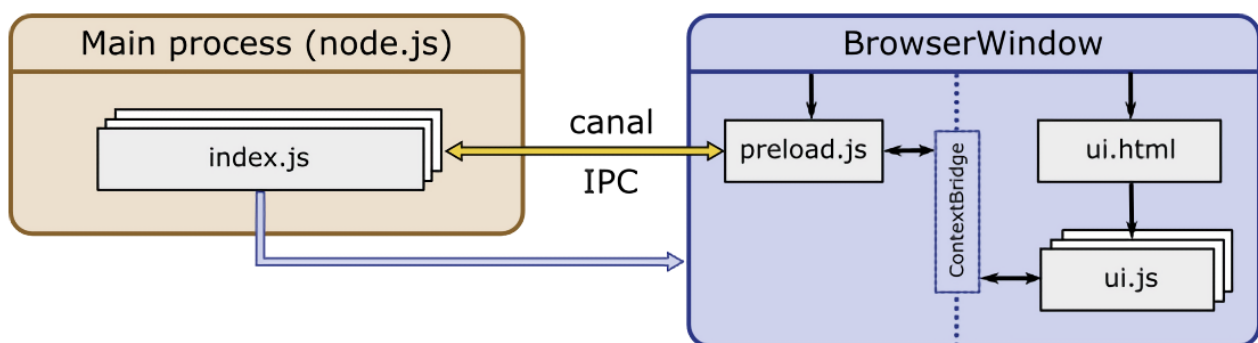
# ВОЗМОЖНОСТИ

## IPC & API

### Inter-process communication

Коммуникация между electron backend'ом (**main process**) и frontend'ом (**renderer process**) осуществляется с помощью межпроцессного взаимодействия (**IPC**).

Чтобы не раскрывать все методы из `ipcRenderer` полностью, используется скрипт-посредник (**preload**).



### Пример: удаление книги из библиотеки

main – `app/electron/main.js`:

```
const io = require("./io");
// <...>

// listen to file delete event
ipcMain.on("app:on-file-delete", (event, file) => {
  io.deleteFile(file.path);
});

// <...>
```

**preload** – `app/electron/preload.js`:

```
const { contextBridge, ipcRenderer } = require("electron");
const fs = require("fs");
const i18nextBackend = require("i18next-electron-fs-backend");
const Store = require("secure-electron-store").default;
const ContextMenu = require("secure-electron-context-menu").default;
const SecureElectronLicenseKeys = require("secure-electron-license-keys");

// Create the electron store to be made available in the renderer process
const store = new Store();

// whitelist channels
const validChannels = [
  // <...>
  "app:file-is-deleted",
  // <...>
];

// Expose protected methods that allow the renderer process to use
// the ipcRenderer without exposing the entire object
contextBridge.exposeInMainWorld("api", {
  i18nextElectronBackend: i18nextBackend.preloadBindings(ipcRenderer, process),
  store: store.preloadBindings(ipcRenderer, fs),
  contextMenu: ContextMenu.preloadBindings(ipcRenderer),
  licenseKeys: SecureElectronLicenseKeys.preloadBindings(ipcRenderer),

  send: (channel, data) => {
    if (validChannels.includes(channel)) {
      ipcRenderer.send(channel, data);
    }
  },
  // <...>
});
```

**renderer** – `app/src/pages/library/LibraryListCard.jsx`:

```
// <...>

const Card = ({ /* <...> */ }) => {
  const { files, setFiles } = useContext(AppContext);

  const handleDelete = (e, file) => {
    e.preventDefault();

    const fileIndex = files.findIndex((fileItem) => fileItem === file);
    setFiles.remove(fileIndex);

    window.api.send("app:on-file-delete", file);
    // Chokidar (from `io.js`) then triggers file update
  };
  // <...>

  return (
    /* <...> */

    <Button
      className="btn-danger"
      compact
      leftIcon={<Trash strokeWidth={1.25} color="var(--clr-danger-000)" />}
      divider={true}
      onClick={(e) => handleDelete(e, file)}
    >
      Delete
    </Button>

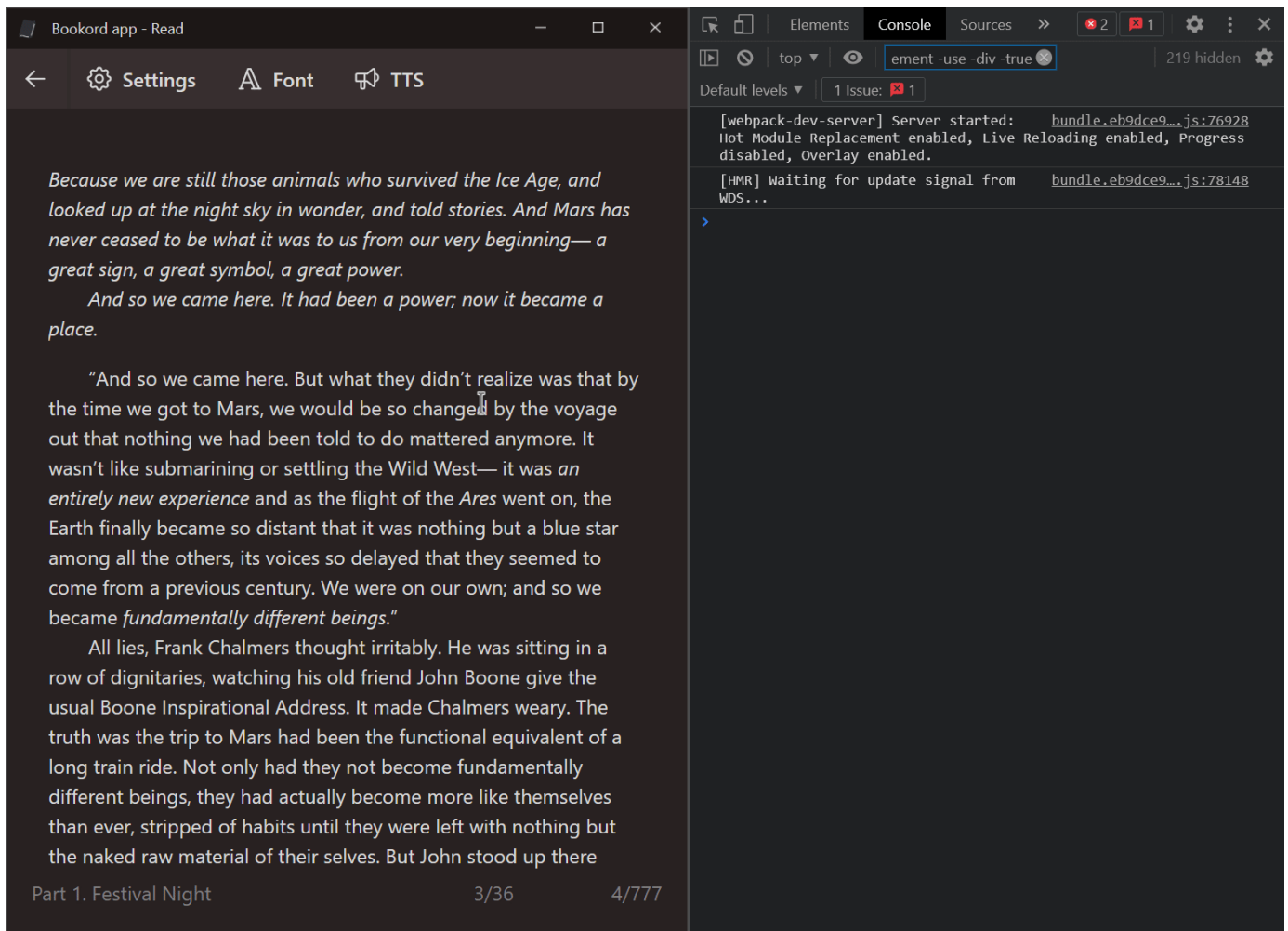
    /* <...> */
  );
}
```

## DeepL translation API

Используя API ключ от DeepL Translator, делается запрос с выделенным текстом, а в ответ DeepL отправляет уже переведенный текст

Отображение tooltip'a пока не доделано





Ссылка на gif

Из `app/src/pages/read/ReadContextMenu.jsx`:

```
// < ... >

const ReadContext = () => {
  const [opened, setOpened] = useDisclosure(false);
  const [contextMenuEvent, setContextMenuEvent] = useState(null);
  const [toTooltip, setToTooltip] = useState({
    target: null,
    originalText: "",
    translatedText: "",
    targetLang: "",
    sourceLang: "",
    selection: null,
  });

  const handleTranslate = () => {
    const targetLang = "RU";
    const originalText = contextMenuEvent.selectedText;
    const selection = contextMenuEvent.selection;

    fetch("https://api-free.deepl.com/v2/translate", {
      method: "POST",
      headers: {
        Host: "api-free.deepl.com",
        "User-Agent": "YourApp",
        Accept: "*/*",
        "Content-Length": "100",
        "Content-Type": "application/x-www-form-urlencoded",
      },
    });
  };
};
```

```

    },
    body: new URLSearchParams({
      auth_key: SECRET.DEEPL,
      text: originalText,
      target_lang: targetLang,
    }).toString(),
  })
  .then((response) => {
    if (response.ok) {
      return response.json();
    }
    throw new Error("Something went wrong");
  })
  .then((response) => {
    setToTooltip(() => ({ ...toTooltip, target: null }));
    const { text: translatedText, detected_source_language: sourceLang } =
      response.translations[0];

    const target = contextMenuEvent.event.path[0];

    setToTooltip(() => ({
      target,
      originalText,
      translatedText,
      targetLang,
      sourceLang,
      selection,
    }));
    setOpenedTooltip(true);
    console.log(sourceLang + "→" + targetLang, translatedText);
  })
  .catch((error) => {
    console.log(error);
  });
};

const [openedTooltip, setOpenedTooltip] = useState(true);

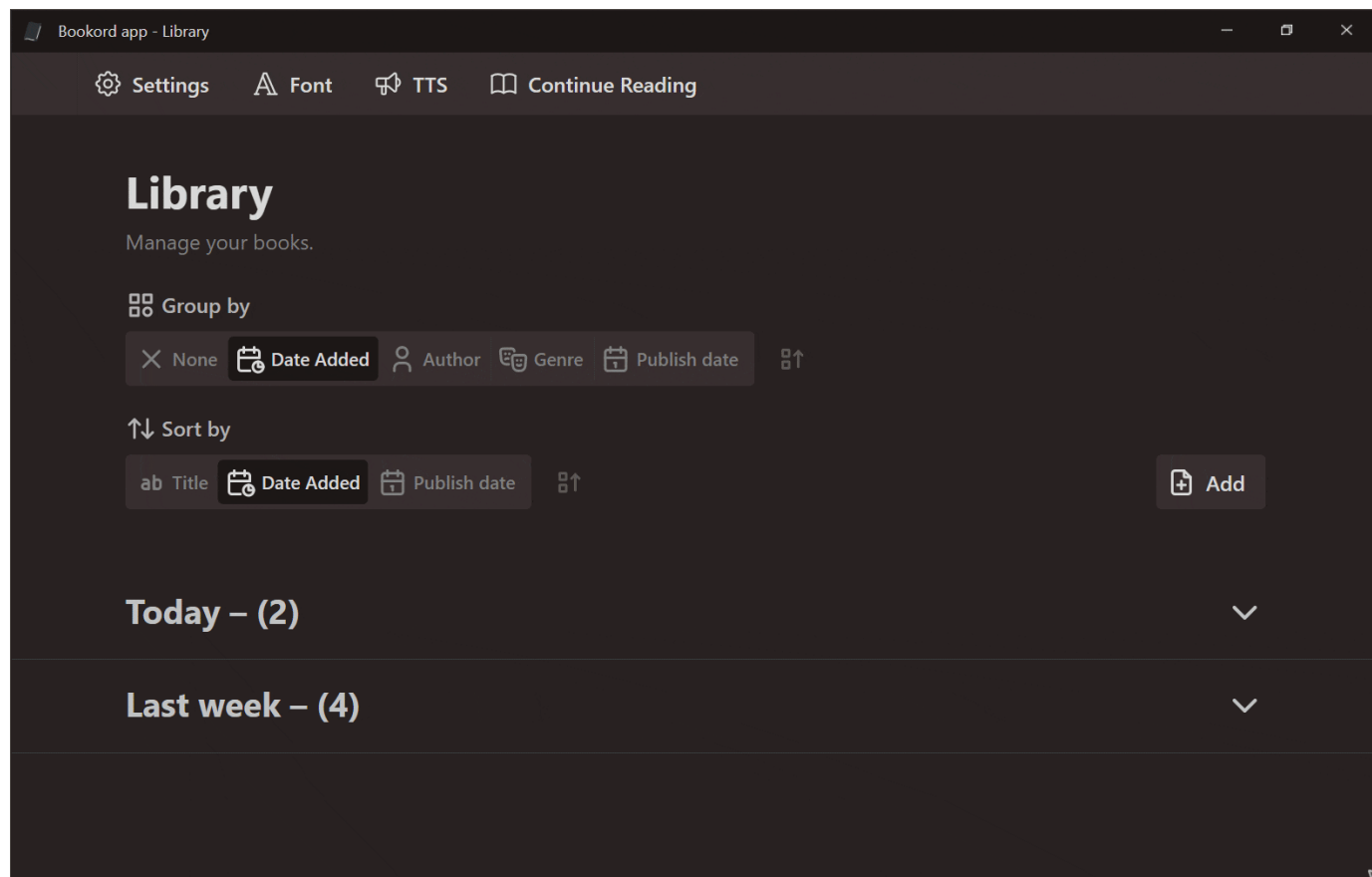
return (
  <
    <ContextMenu
      opened={opened}
      setOpened={setOpened}
      setContextMenuEvent={setContextMenuEvent}
      position={undefined}
    >
      { /* < ... > */ }
      <Button
        onClick={handleTranslate}
        compact
        leftIcon={<Language strokeWidth={1.25} color="var(--clr-primary-100)" /> }
      >
        Translate
      </Button>
    </ContextMenu>

    {toTooltip.target && <PortalTooltip opened={openedTooltip} toTooltip={toTooltip}
  } />
  </>
);
};

```

# Доступность

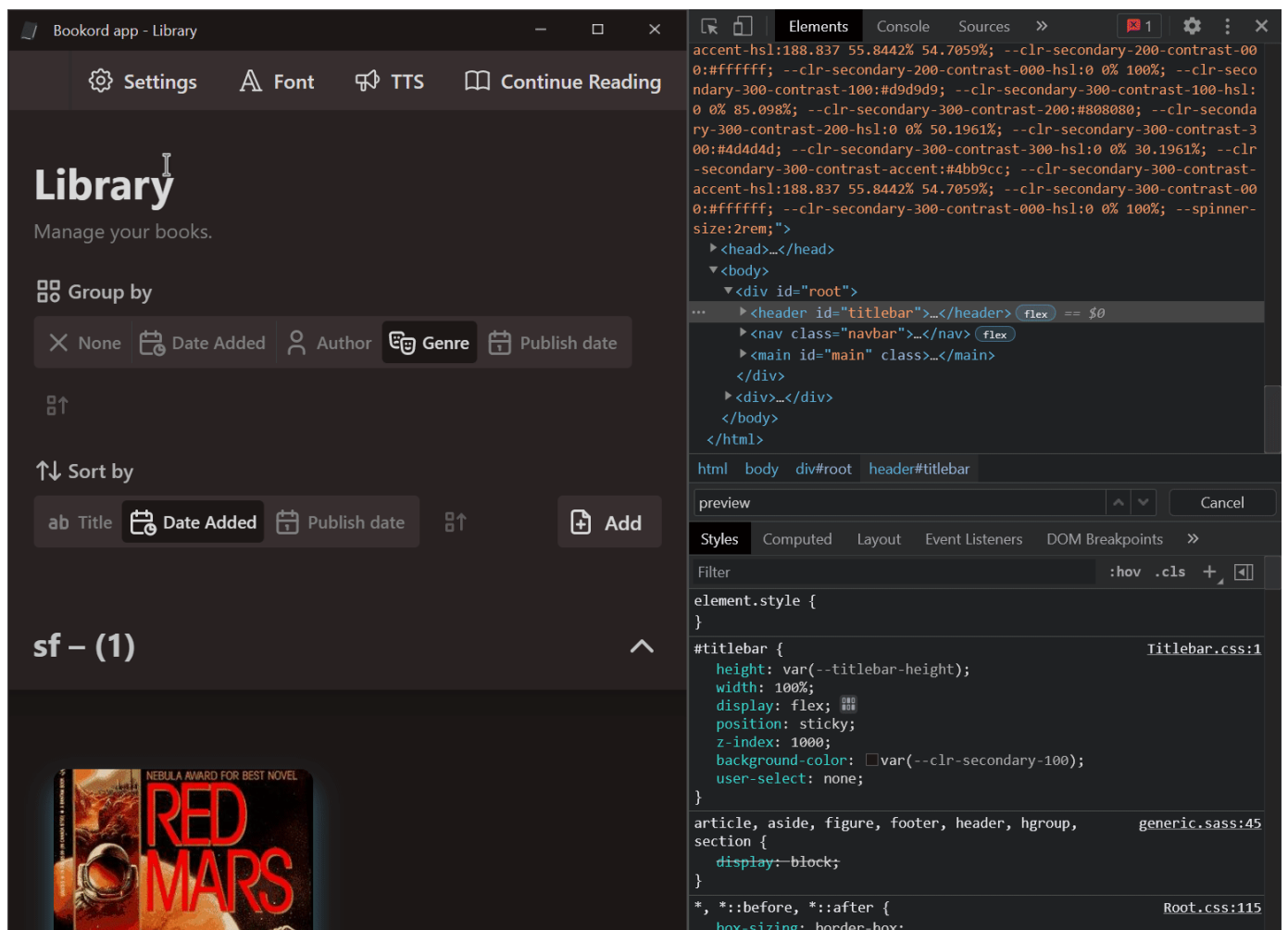
## Навигация всего приложения с помощью клавиатуры



Ссылка на gif

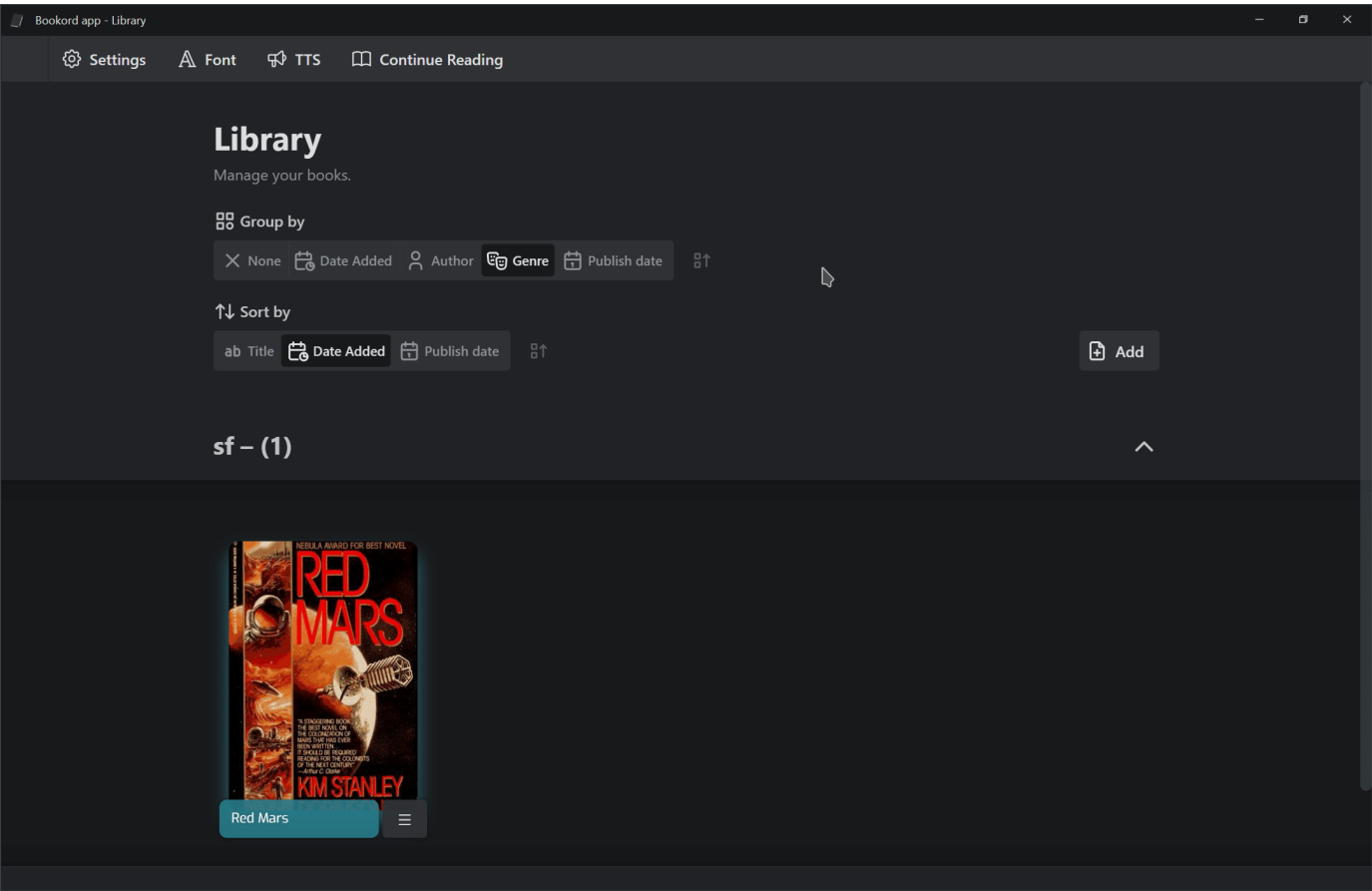
## HTML структура

Используются семантические HTML теги: верхняя часть страницы – `header`; панель навигации под ней – `nav`; основная часть – `main`, внутри которого `section`, с оглавлениями `h1`, `h2`...



Ссылка на gif

## Адаптивность



Ссылка на gif

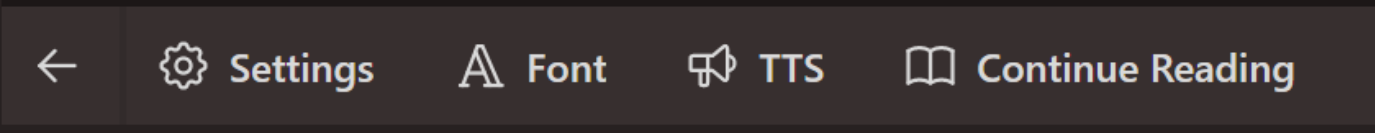
## Text-To-Speech

Пока не реализовано

## Иконки

## Использование `tablers-icon-react`

Панель навигации



Из `app/src/core/Navbar.jsx`:

```
// <...>
import { ArrowLeft, Settings, /* <...> */ } from "tabler-icons-react";

const Navbar = ({ /* <...> */ }) => {
  // <...>

  return (
    <
      <nav className="navbar">
        <ul className="navbar-list">
          <li
            className="navbar-item"
            // @ts-ignore
            style={!isBtnBackVisible ? invisibleStyle : null}
          >
            <Button
              isVisible={isBtnBackVisible}
              to={ROUTES.LIBRARY}
              title="Back to Library"
              isIconOnly={true}
            >
              <ArrowLeft strokeWidth={1.5} />
            </Button>
          </li>
          <Divider size="md" orientation="vertical" />
          <li className="navbar-item">
            <Button
              to={ROUTES.SETTINGS}
              title="Application Settings"
              leftIcon={<Settings strokeWidth={1.5} />}
            >
              Settings
            </Button>
          </li>
        </ul>
      </nav>
    </>
  );
}
```

## Темизация

Умный color picker автоматически подбирает подцвета не лишая, при этом, возможности тонко настроить их вручную.

## Цвет акцента



## Color

Repaint the app!

### Accent color

TODO.

 #277b8a



Show Advanced



### Primary color

TODO.

 #ffffff




Show Advanced



### Secondary color

TODO.

 #000000

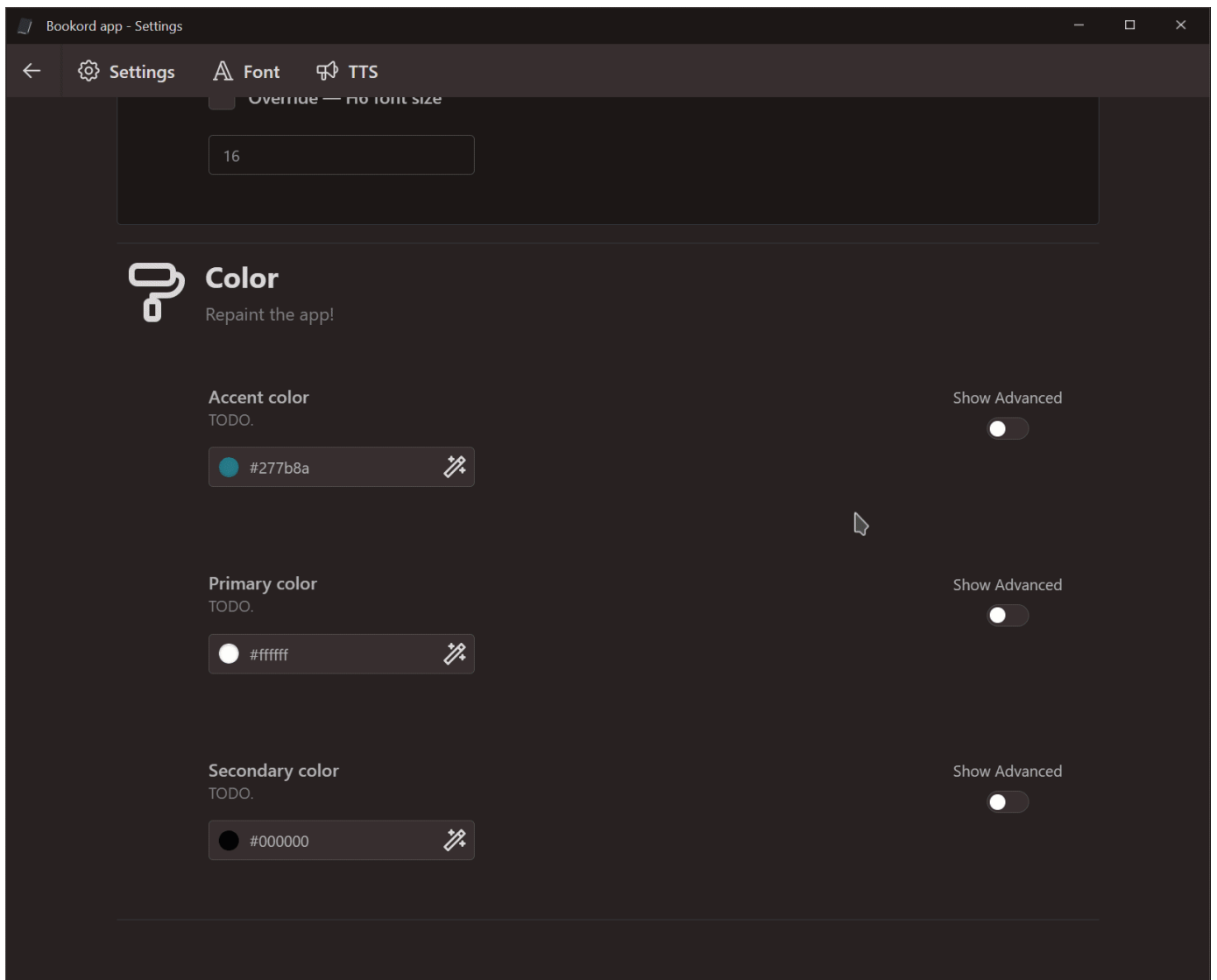


Show Advanced



Ссылка на gif

## Главные и вторстепенные цвета

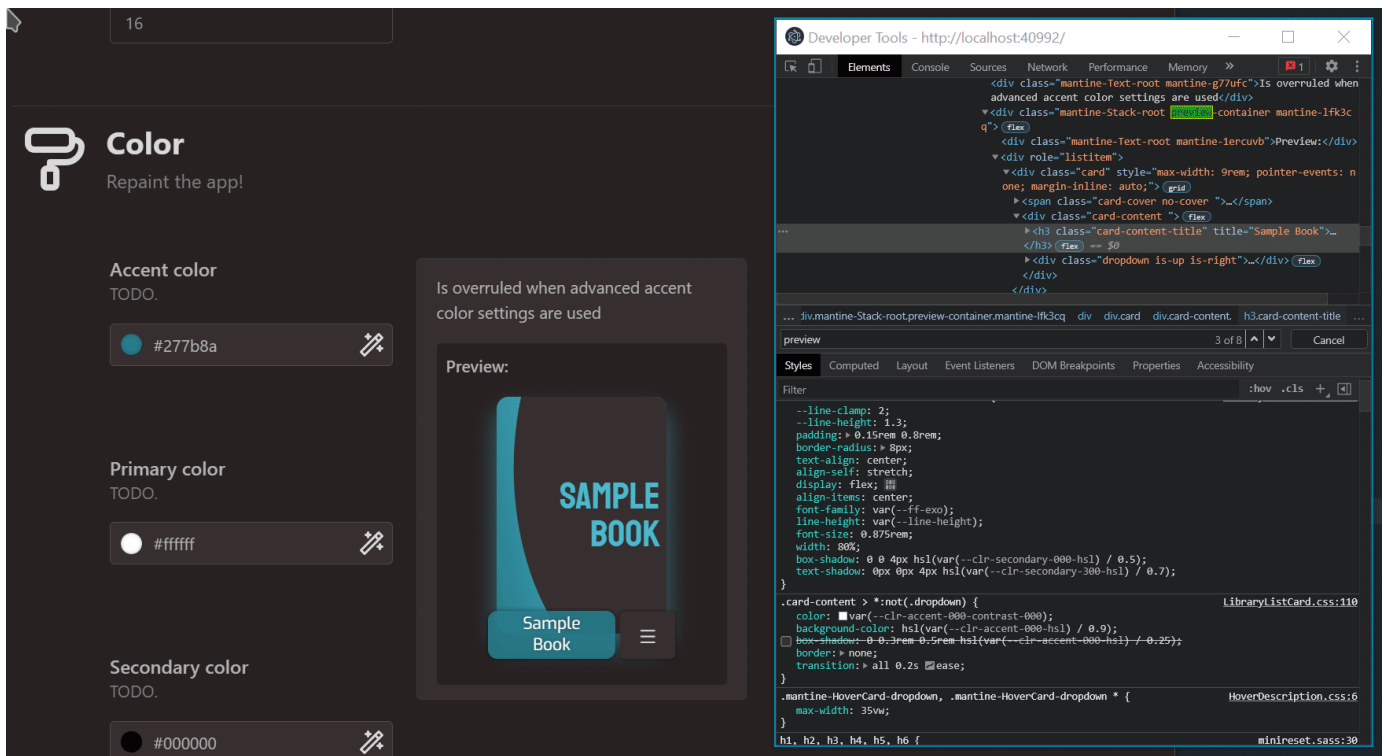


Ссылка на gif

## Реализация: изменение цвета

Все цвета находятся в CSS переменных и они меняются за счет метода `CSSStyleDeclaration.setProperty()`





Ссылка на gif

Из `app/src/pages/library/LibraryListCard.css`:

```
/* <...> */
.card-content > *:not(.dropdown) {
  color: var(--clr-accent-000-contrast-000);
  background-color: hsl(var(--clr-accent-000-hsl) / 0.9);
  border: none;

  transition: all 0.2s ease;
}
/* <...> */
```

Из `app/src/utils/cssColors.js` функция `updateCssVar`:

```
const updateCssVar = (setting) => {
  if ("theme" in setting) {
    if (!setting.disabled) {
      // <...>
      if (setting.type === "colorInput") {
        document.documentElement.style.setProperty(
          setting.theme.cssVar,
          setting.value
        );
        // Generates hsl variable version for color
        const color = tinycolor(setting.value);
        const { h, s, l } = color.toHsl();
        const hslString = `${h} ${s * 100}% ${l * 100}%`;

        document.documentElement.style.setProperty(
          setting.theme.cssVar + "-hsl",
          hslString
        );
      }
      // <...>
    }
  }
}
```

## Реализация: подбор подцвета

Из `app/src/pages/settings/SettingsSections.jsx` функция `updateSettings`:

```
const updateSettings = (settingKey, value, parentSettingKey, valueKey = "value") => {
  // <...>
  settingOrSubsetting.theme.controlledSettings.forEach((controlledSettingObj) => {
    const { subsettingKey, h, s, l } = controlledSettingObj;
    const controlledSetting = {
      ...settings[parentSettingKey].subsettings[subsettingKey],
    };
    const color = tinycolor(value);
    color.isDark()
      ? color.lighten(l).saturate(s).spin(h)
      : color.darken(l).desaturate(s).spin(-h);

    controlledSetting.value = color.toString();
    controlledSetting.defaultValue = color.toString();
    updatedSubsettings[subsettingKey] = controlledSetting;
  });
  // <...>
}
```

Изменение подцветов

ИЗ `app/src/constants/defaultSettings.json`:

```
"accentColor": {
  "name": "Accent color",
  // <...>
  "subsettings": {
    "accentColorGlobal": {
      // <...>
      "value": "#277b8a",
      "theme": {
        "cssVar": "--clr-accent-global",
        "isControlledApplied": true,
        "controlledSettings": [
          {
            "subsettingKey": "accentColor000",
            "h": 0,
            "s": 0,
            "l": 0
          },
          {
            "subsettingKey": "accentColor100",
            "h": 0,
            "s": 0,
            "l": 20
          }
        ]
      }
    }
  },
  "accentColor000": {
    // <...>
    "value": "#277b8a",
    "theme": {
      "cssVar": "--clr-accent-000"
    }
  },
  "accentColor100": {
    // <...>
    "value": "#4bb9cc",
    "theme": {
      "cssVar": "--clr-accent-100"
    }
  }
}
```

Параметры `h`, `s`, `l` для подцветов, описывающие их отличие от их родительского цвета