HTML Editor

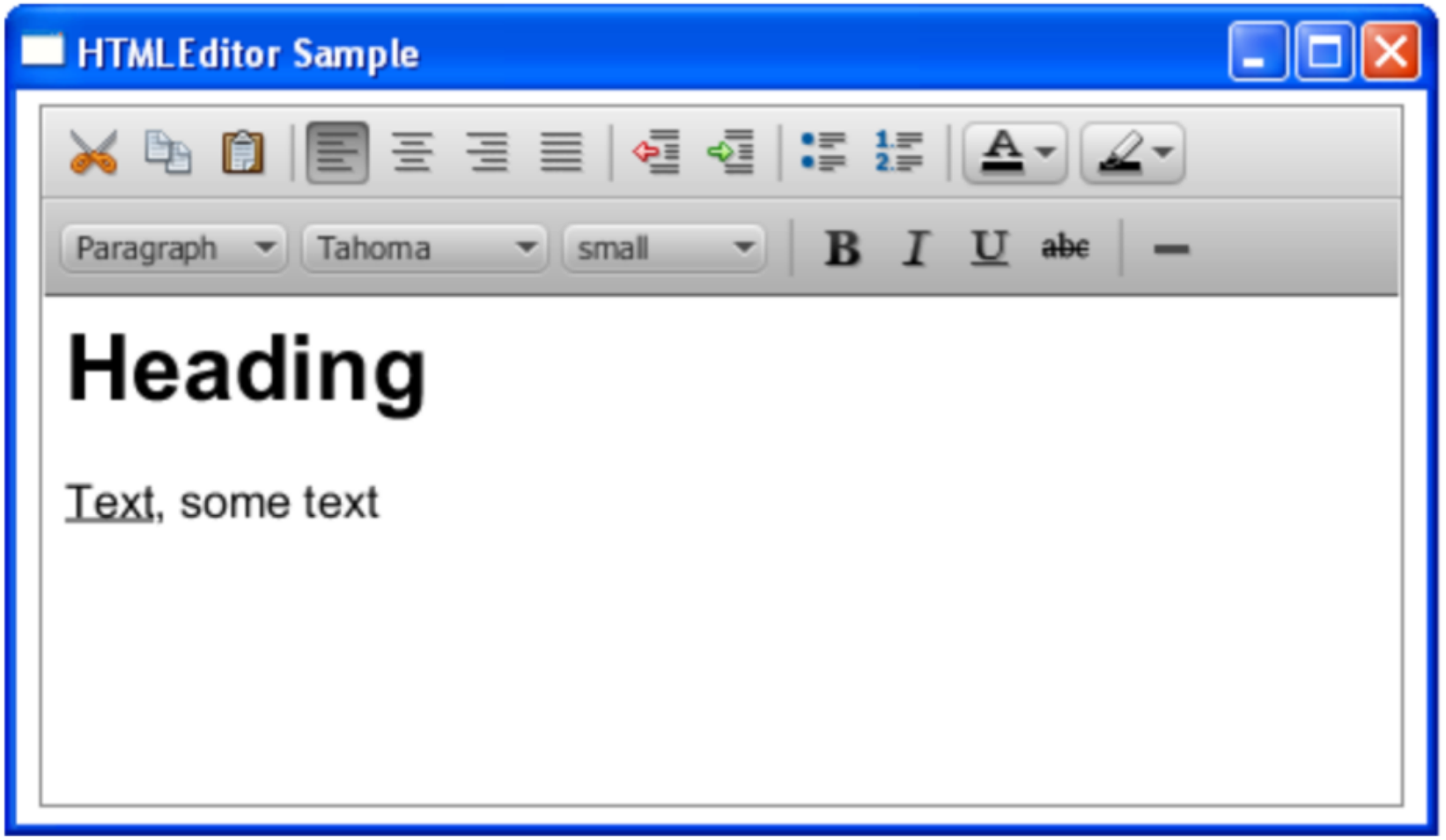
BY: PROFESSOR WERGELES

Introduction

- In this lecture, you learn how to edit text in your JavaFX applications by using the embedded HTML editor
- The HTMLEditor control is a full functional rich text editor
  - Its implementation is based on document editing feature of HTML5 and includes the following editing functions:
    - Text formatting including bold, italic, underline, and strike though styles
    - Paragraph settings such as format, font family, and font size
    - Foreground and background colors
    - Text indent
    - Bulleted and numbered lists
    - Text alignment
    - Adding a horizontal rule
    - Copying and pasting text fragments

Introduction

Figure 19-1 HTML Editor



Introduction

- HTMLEditor class presents editing content in the form of an HTML string
- For example, content typed in editor in Figure 19-1 is presented by the following string:
  - "<html>
    - <head></head>
    - <body contenteditable="true">
      - <h1>Heading</h1>
      - <u>Text</u>, some text
    - </body>
  - </html>."
- Because the HTMLEditor class is an extension of the Node class, you can apply visual effects or transformations to its instances

Adding an HTML Editor

- Like any other UI control, HTMLEditor component must be added to scene so it can appear in app
- You can add it directly to scene as shown in Example 19-1
  - or through a layout container as done in other examples

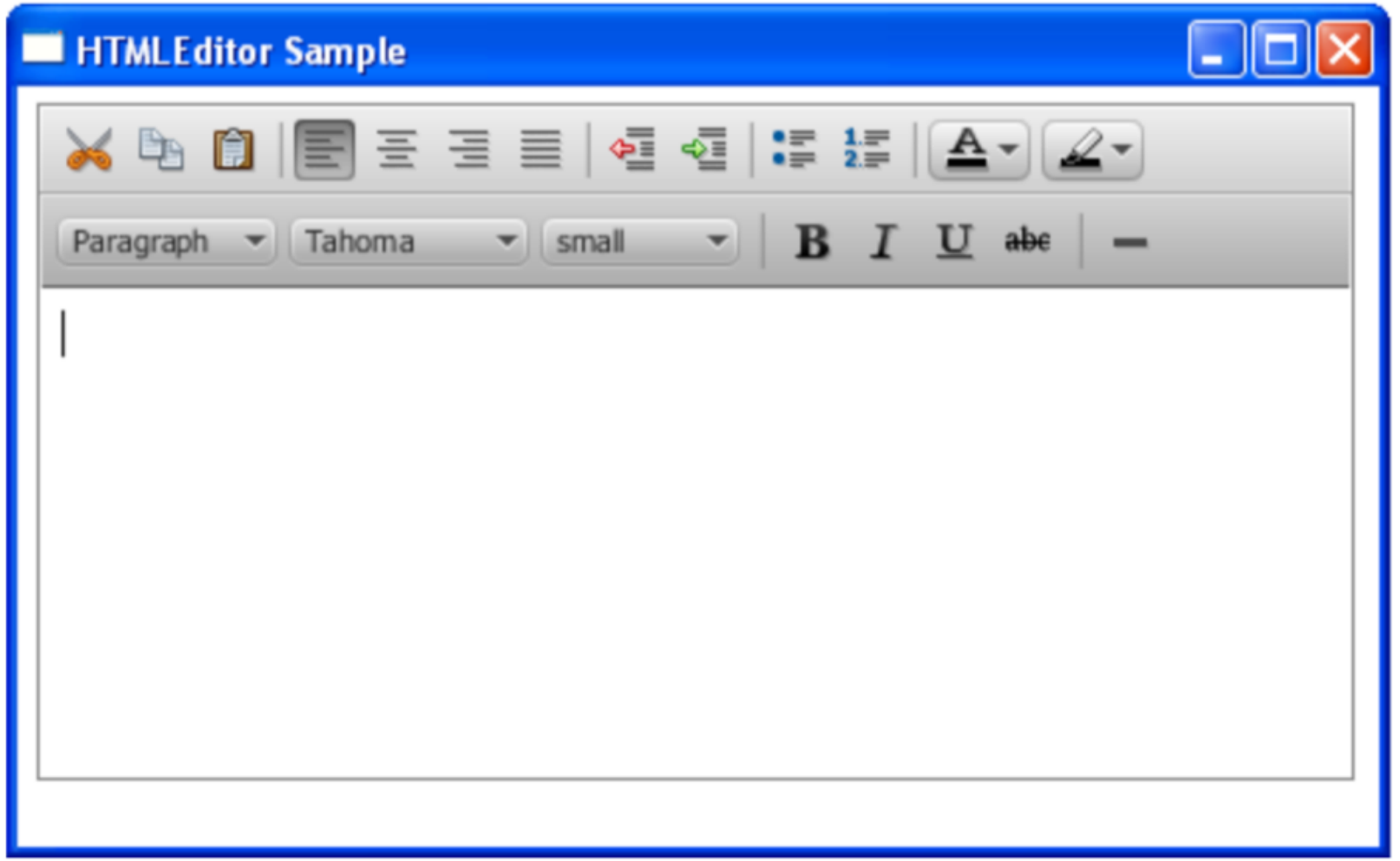
Example 19-1 Adding an HTML Editor to a JavaFX Application

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.web.HTMLEditor;
import javafx.stage.Stage;

public class HTMLEditorSample extends Application {

    @Override
    public void start(Stage stage) {
        stage.setTitle("HTMLEditor Sample");
        stage.setWidth(400);
        stage.setHeight(300);
        final HTMLEditor htmlEditor = new HTMLEditor();
        htmlEditor.setPrefHeight(245);
        Scene scene = new Scene(htmlEditor);
        stage.setScene(scene);
        stage.show();
    }
```

Adding an HTML Editor

- Compiling and running this code produces window shown in Figure 19-2

Figure 19-2 Initial View of the HTMLEditor Component



Adding an HTML Editor

- Formatting toolbars are provided in implementation of component
  - You cannot toggle their visibility
- However, still can customize appearance of editor by applying CSS style, as shown in Example 19-2

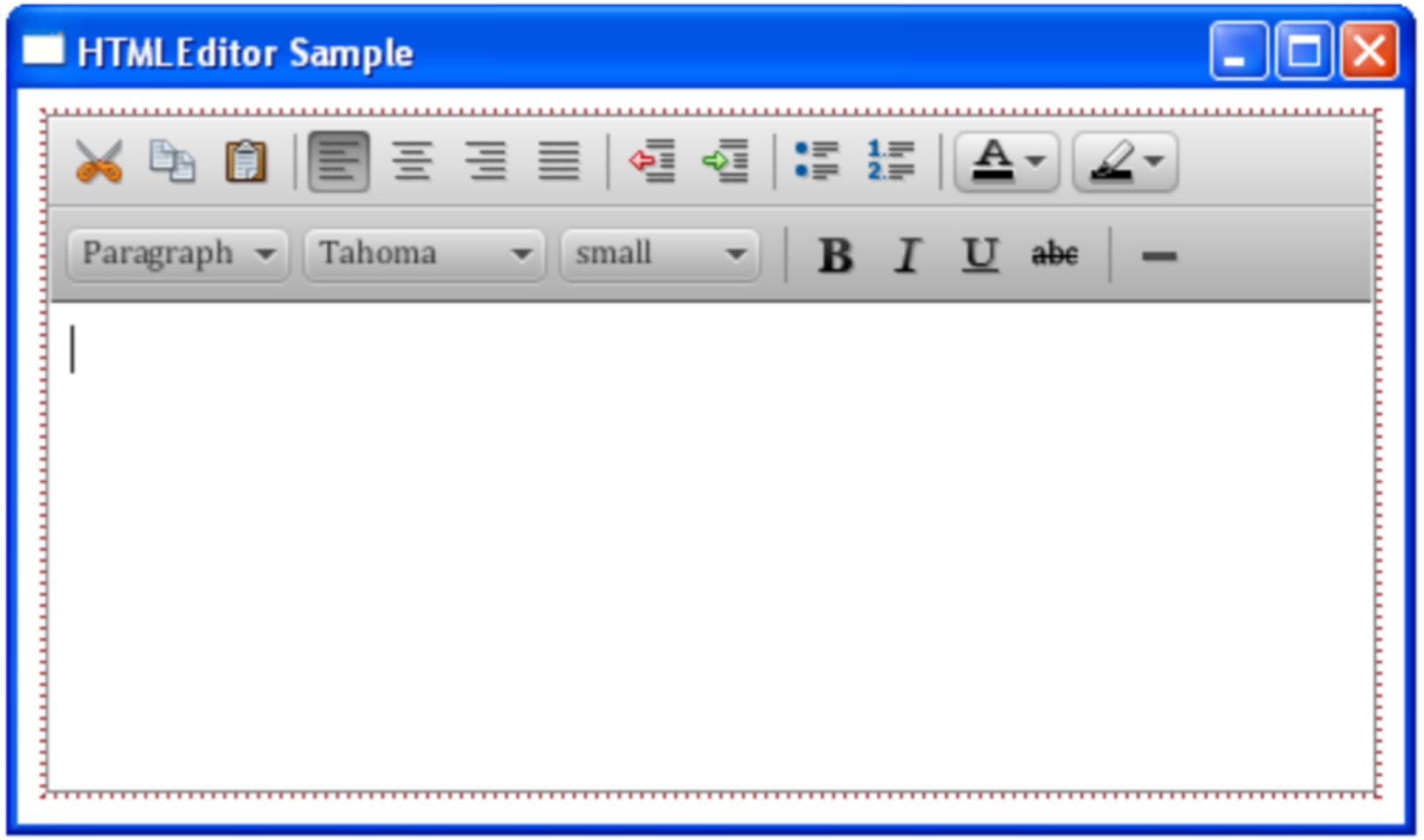
Example 19-2 Styling the HTMLEditor

```
htmlEditor.setStyle(
    "-fx-font: 12 cambria;"
    + "-fx-border-color: brown; "
    + "-fx-border-style: dotted;"
    + "-fx-border-width: 2;"
);
```

Adding an HTML Editor

- When this code is added, the editor changes, as shown in Figure 19-3
- The applied style changes border of component and font of formatting toolbars

Figure 19-3 Alternative View of the HTMLEditor Component



Adding an HTML Editor

- HTMLEditor class provides method to define content that appears in editing area when application starts
- Use setHtmlText method, as shown in Example 19-3, to set the initial text for editor

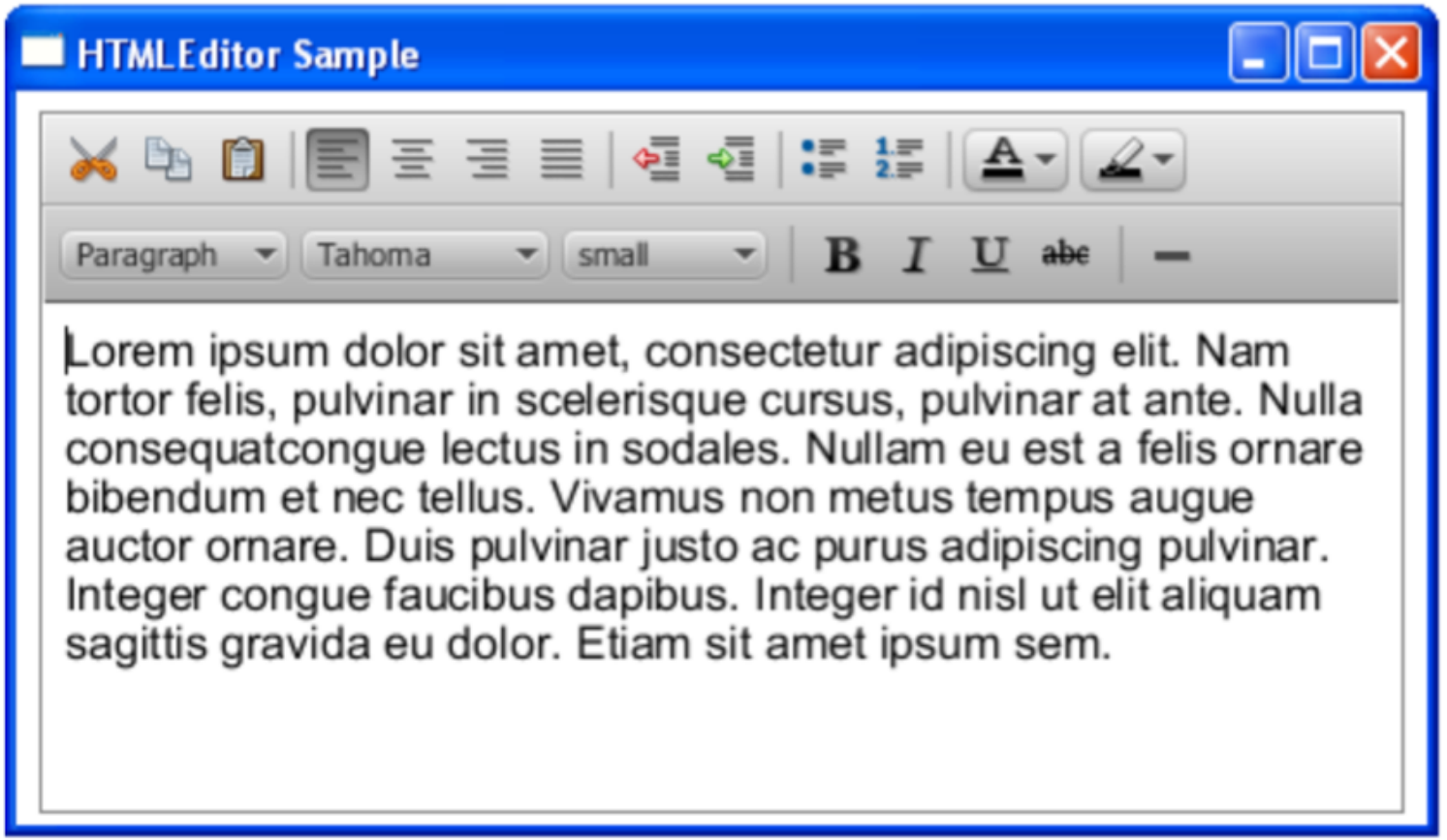
Example 19-3 Setting the Text Content

```
private final String INITIAL_TEXT = "<html><body>Lorem ipsum dolor sit "
    + "amet, consectetur adipiscing elit. Nam tortor felis, pulvinar "
    + "in scelerisque cursus, pulvinar at ante. Nulla consequat"
    + "congue lectus in sodales. Nullam eu est a felis ornare "
    + "bibendum et nec tellus. Vivamus non metus tempus augue auctor "
    + "ornare. Duis pulvinar justo ac purus adipiscing pulvinar. "
    + "Integer congue faucibus dapibus. Integer id nisl ut elit "
    + "aliquam sagittis gravida eu dolor. Etiam sit amet ipsum "
    + "sem.</body></html>";

htmlEditor.setHtmlText(INITIAL_TEXT);
```

Adding an HTML Editor

- Figure 19-4 demonstrates text editor with text set by using the setHTMLText method
- You can use HTML tags in this string to apply specific formatting for initially rendered content

Figure 19-4 HTMLEditor with the Predefined Text Content



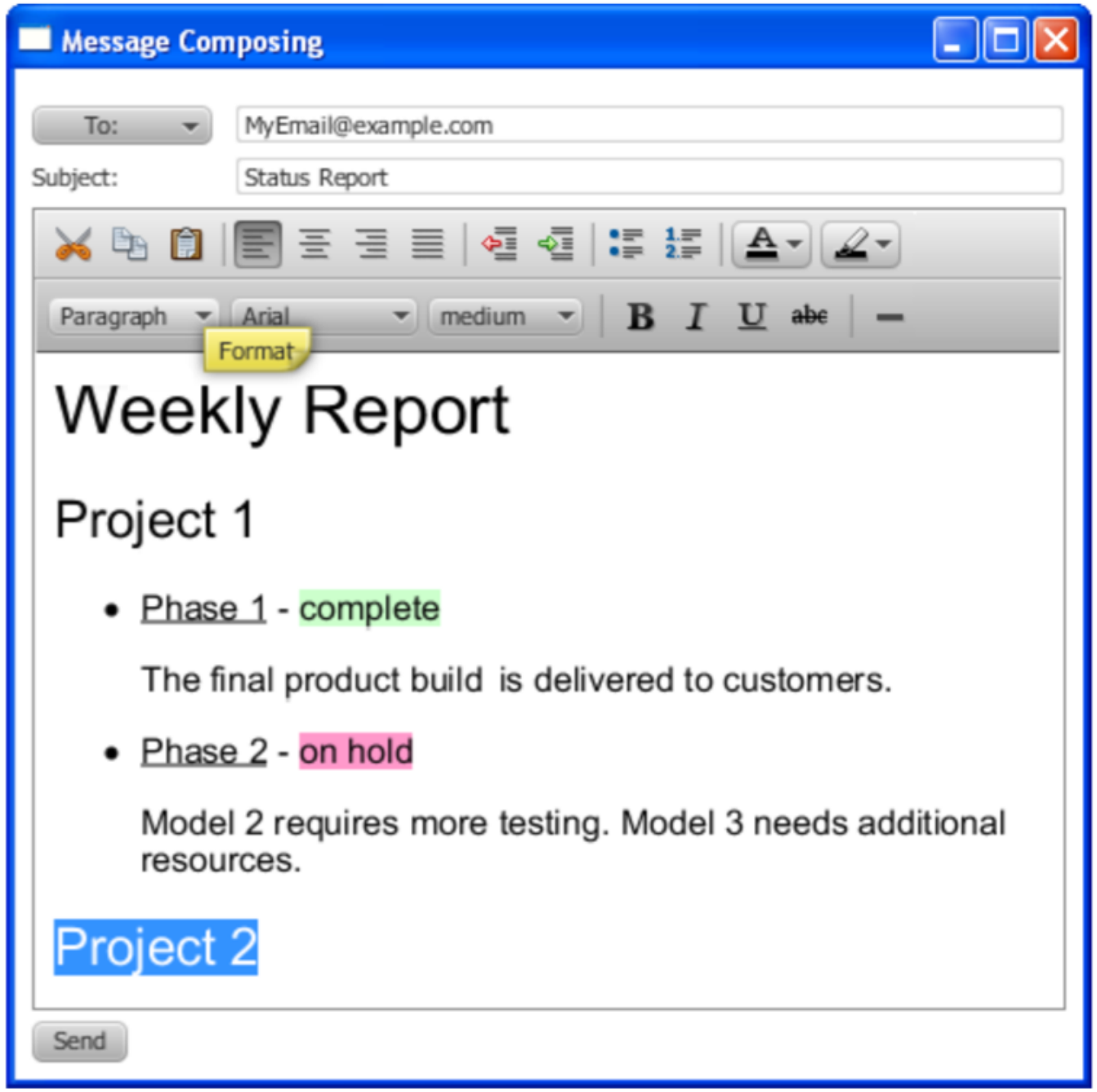
Using HTML Editor to Build UI

- You can use HTMLEditor control to implement typical user interfaces (UIs) in JavaFX applications
  - For example, you can implement instant messenger services, email clients, or even content management systems
- This example presents UI of message composing window found in many email client applications
  - UI includes choice box to select a type of recipient, two text fields to enter email address and subject of message, label to indicate subject field, the editor, and Send button
  - UI controls arranged on application scene by using Grid and VBox layout containers
  - When you compile and run application, window shown in Figure 19-5 shows output of application when a user is composing a weekly report

Using HTML Editor to Build UI

- Can set specific width/height values for HTMLEditor object by calling setPrefWidth or setPrefHeight methods, or leave its width/height unspecified
- Example 19-4 specifies height of the component
  - Width is defined by VBox layout container
  - When text content exceeds height of editing area, vertical scroll bar appears

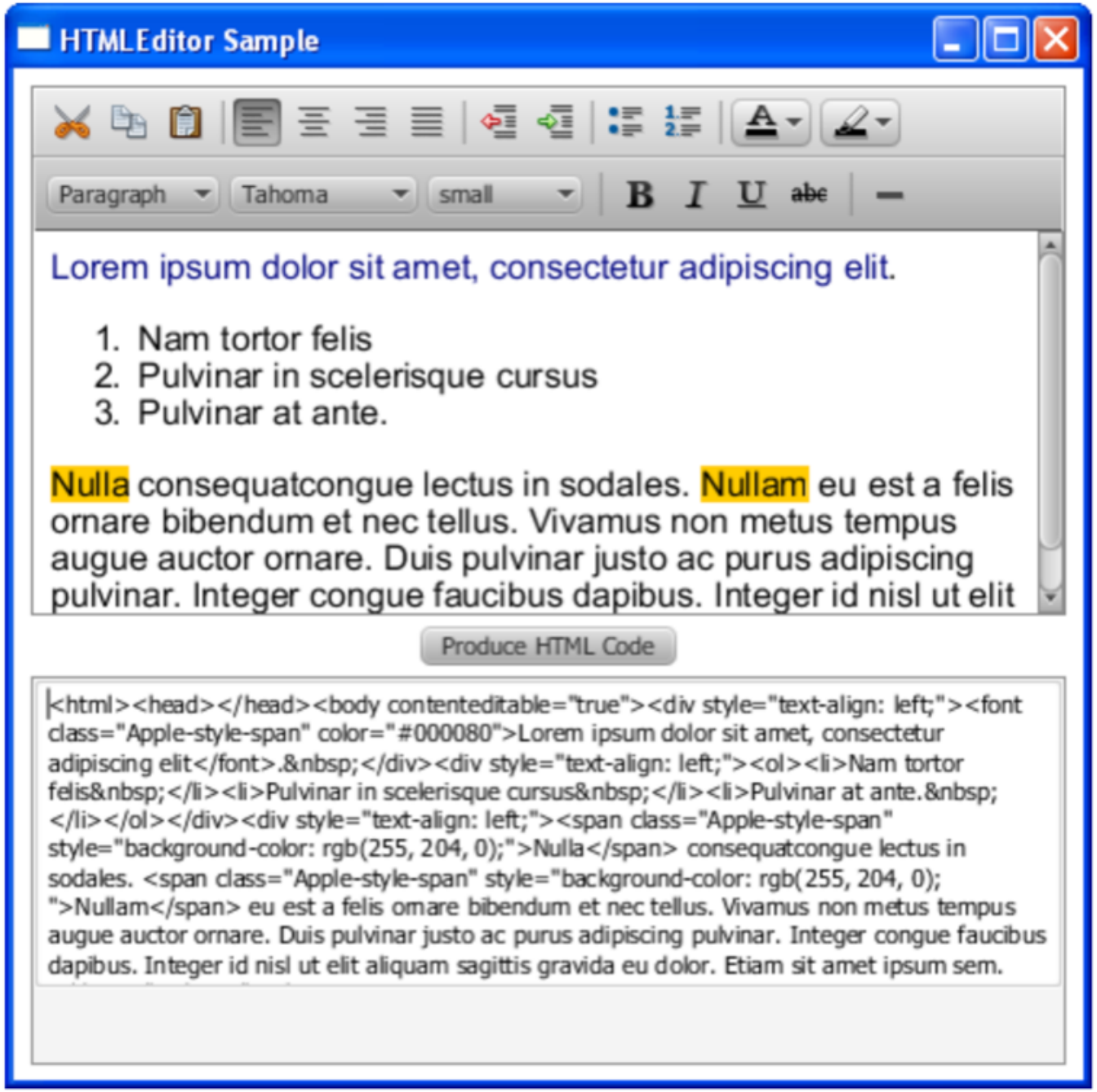
Figure 19-5 Email Client User Interface



Obtaining HTML Content

- With JavaFX HTMLEditor control, you can edit text and set initial content
  - In addition, you can obtain entered and edited text in HTML format
- The getHTMLText method called on the HTMLEditor object derives edited content and presents an HTML string
  - This information is passed to text area, so that you can observe, copy, and paste the produced HTML code
- Figure 19-6 shows HTML code of the text being edited in HTMLEditor

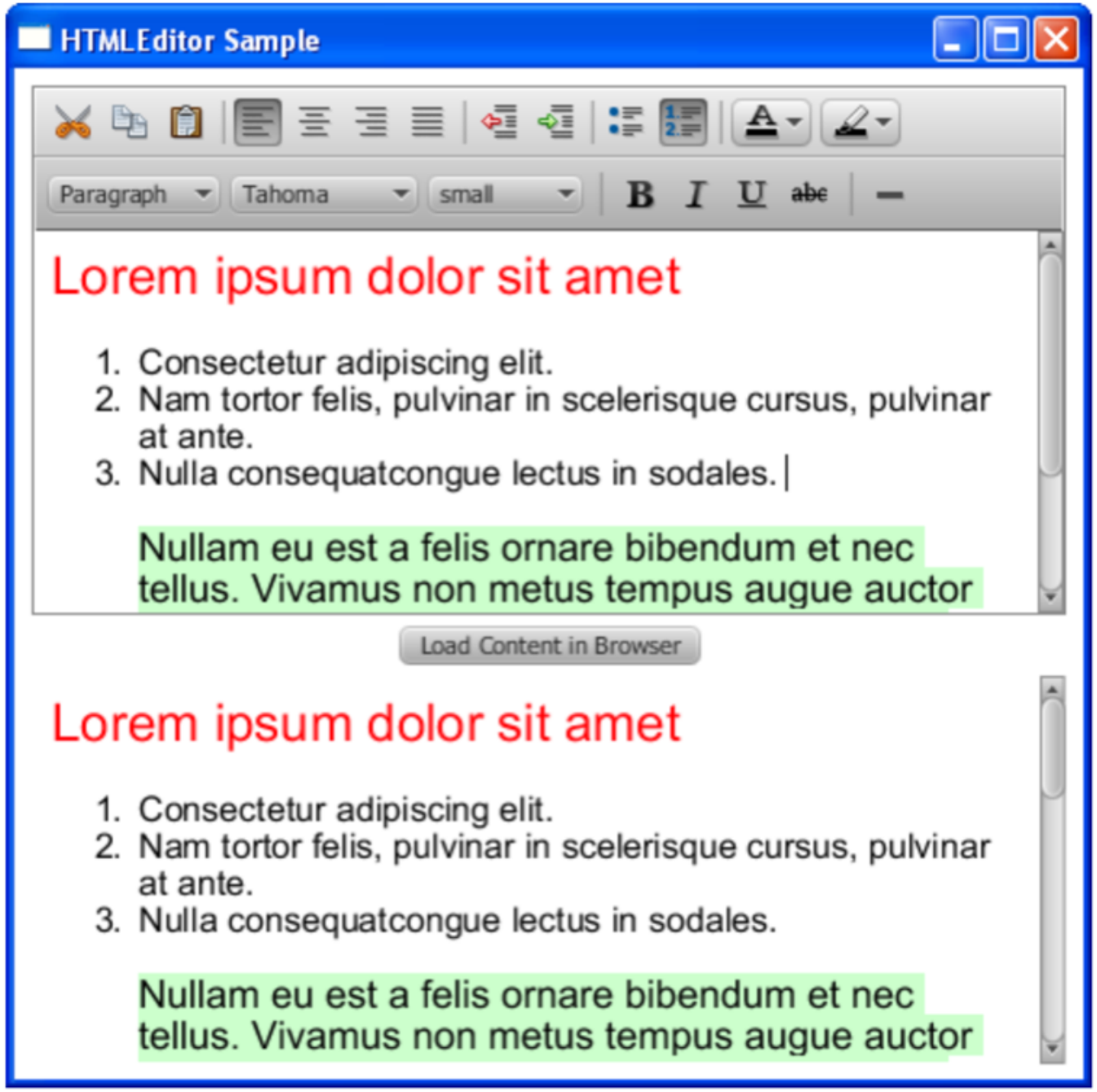
Figure 19-6 Obtaining the HTML Content



Obtaining HTML Content

- Similarly, you can obtain HTML code and save it in file or send it to WebView object to synchronize content in editor and embedded browser
- HTML code received from htmlEditor component is loaded in WebEngine object that specifies content for embedded browser
- Each time user clicks Load Content in Browser button, edited content is updated in browser
- Figure 19-7 demonstrates this in action

Figure 19-7 Loading Content in a Browser



Coding Exercise

- Go and code Document Editor 1 and 2