

Theming und UI-Modernisierung

Theming und der Theme Builder · Visuelle Hierarchie und gefahrloser Feinschliff

Theming und der Theme Builder

Wenden Sie ein integriertes Theme an, sehen Sie, wo Theme-Dateien und Stiloptionen ihren Platz haben, und nutzen Sie den Theme Builder auf Einsteigerniveau – ohne die Geschäftslogik anzufassen.



Lernziel der Lektion

Wenden Sie ein integriertes Wisej.NET-**Theme** an, verstehen Sie, wo Theme-Dateien und Stiloptionen ihren Platz haben, und nutzen Sie den Theme Builder auf praxisnahem Einsteigerniveau – alles, ohne die Geschäftslogik anzutasten.

1. Theming und Geschäftslogik

Theming ändert, *wie* die Anwendung aussieht. Geschäftslogik ändert, *was* sie tut. In einer Geschäftsanwendung sollten Sie Farben, Abstände, Karten, Header und Navigation verbessern können, ohne Services, Repositories, Validierungsregeln oder Ticket-Abläufe neu zu schreiben.

Bereich	Entscheidung zur UI-Modernisierung	<i>Nicht</i> neu schreiben
Theme	Wählen Sie ein integriertes Theme wie Bootstrap-4, BootstrapDark-4, Blue-1 oder Classic-2.	Ticket-Service, Speicherlogik, Validierungsregeln.
Abstände	Verwenden Sie einheitliche Innen-, Außen- und Kartenabstände.	Datenmodellklassen oder Repository-Methoden.
Gruppierung	Fassen Sie zusammengehörige Befehle und Informationen in Karten / Panels zusammen.	Navigationsablauf oder Berechtigungsprüfungen.
Beschriftungen	Formulieren Sie Überschriften, Feldbeschriftungen und Hilfetexte klar.	Die eigentlichen Berechnungen oder gespeicherten Ticket-Datensätze.
Befehle	Platzieren Sie primäre Buttons dort, wo Benutzer sie erwarten; halten Sie die Namen vorhersehbar.	Geschäftsregeln in Event-Handlern.

2. Theme-Dateien, Schriften, Bilder und Stiloptionen

- Ein Theme gibt der App einen einheitlichen Grundstil für die gängigen Controls.
- Schriften sollten gut lesbar und einheitlich sein – vermeiden Sie zu viele Größen oder Schriftfamilien.

- Bilder und Icons sollten den Arbeitsablauf unterstützen und keine Dekoration hinzufügen, die den Benutzer verwirrt.
- Stiloptionen sollten die Klarheit verbessern: aktiver Navigationszustand, klare Abschnittsüberschriften, gut lesbare Buttons.
- Behalten Sie ein einziges visuelles System bei – gleiche Button-Größen, Kartenabstände, Beschriftungsmuster und Farbbedeutungen auf allen Seiten.

3. Das Konzept des Theme Builders

Der Theme Builder ist für Fälle gedacht, in denen die integrierten Themes fast, aber nicht genau dem entsprechen, was das Projekt braucht. Als Einsteiger wenden Sie zuerst ein integriertes Theme an und nehmen dann kleine, praxisnahe Anpassungen vor – Primärfarbe, Hintergründe, Schriftgrößen, Abstände der Controls. Beginnen Sie nicht damit, jede Seite von Hand zu bearbeiten.



Beginnen Sie mit einem integrierten Theme

Wählen Sie zuerst ein integriertes Theme und passen Sie einige Werte an. Greifen Sie erst dann zum Theme Builder, wenn das Basis-Theme Sie nicht ganz ans Ziel bringt.

4. Eine moderne Seitenstruktur

Eine moderne Geschäftsanwendungsseite hat in der Regel einen klaren Header, eine linke Navigation, einen Hauptinhaltsbereich, gruppierte Karten und eine vorhersehbare Anordnung der Befehle – so wissen Benutzer immer, wo sie sind und was als Nächstes zu tun ist.

Header	→	Titel, Modulbezeichnung, Auswahl des aktuellen Themes
Navigation	→	Dashboard · Tickets · Customers · Reports · Settings
Inhaltsbereich	→	Seitenüberschrift + kurze Beschreibung
	→	eine Reihe von Kennzahlenkarten (große Zahlen)
	→	gruppierte Befehle letzte Aktivitäten



Sie sind bereit für die nächste Lektüre

Sie können Theming von Geschäftslogik trennen, ein integriertes Theme anwenden und eine moderne Seitenstruktur beschreiben. Als Nächstes: die Regeln der visuellen Hierarchie und wie Sie einer Seite den Feinschliff geben, ohne Funktionierendes kaputtzumachen.

Visuelle Hierarchie und gefahrloser Feinschliff

Regeln der visuellen Hierarchie, die Masken leicht erfassbar machen, und wie Sie das Layout verfeinern, ohne Binding, Navigation oder Validierung zu beschädigen.



Lernziel der Lektion

Nutzen Sie die Regeln der visuellen Hierarchie, damit Seiten leicht zu erfassen sind, und modernisieren Sie dann das Layout, **ohne** Data Binding, Navigation, Validierung oder Geschäftsregeln zu beschädigen.

1. Regeln der visuellen Hierarchie

Regel	Was sie bedeutet	Beispiel in Modul 7
Den Seitentitel deutlich machen	Der Benutzer sollte sofort wissen, auf welcher Seite er sich befindet.	Überschriften Dashboard, Tickets, Customers, Reports, Settings.
Zusammengehörige Inhalte gruppieren	Verwenden Sie Panels / Karten, damit zusammengehörige Datensätze und Befehle beieinanderbleiben.	Kennzahlenkarten, Karte für die Ticket-Warteschlange, Karte für UI-Einstellungen.
Befehle vorhersehbar anordnen	Platzieren Sie Buttons nahe an den Inhalten, auf die sie sich auswirken.	Create / Edit / Close Ticket unter der Ticket-Warteschlange.
Einheitliche Abstände verwenden	Alle Seiten sollten sich wie dieselbe Anwendung anfühlen.	Gleicher linker Innenabstand, gleiche Kartenbreiten und Zeilenabstände auf allen Seiten.
Aktiven Navigationszustand zeigen	Der ausgewählte Navigations-Button sollte ausgewählt aussehen.	Grüner, ausgewählter Button mit Rahmen auf der aktuellen Seite.

2. Feinschliff, ohne die Logik neu zu schreiben

- Benennen Sie Controls bei Bedarf eindeutig um, vermeiden Sie aber, bestehende Event-Handler zu beschädigen.
- Verschieben Sie zusammengehörige Controls in Panels oder Karten, ohne die Service-Schicht zu ändern.
- Setzen Sie `Dock`, `Anchor` und `AutoSize` bewusst ein, damit die Seite auch bei Größenänderungen bedienbar bleibt.
- Behalten Sie beim Modernisieren von Grids, Listen und Detail-Panels dieselbe Datenquelle und dieselbe Binding-Logik bei.
- Testen Sie nach dem Feinschliff der Oberfläche jeden Navigations-Button und jeden Arbeitsablauf.



Typischer Fehler

Verstecken Sie defekte Logik nicht hinter einem schöneren Theme. Auch eine moderne Oberfläche braucht eine funktionierende Navigation, Validierung, ein korrektes Speichern/Abbrechen-Verhalten, sichere Fehlermeldungen und klare Statusmeldungen.

3. Theme-Einstellungen aus der Geschäftslogik heraushalten

Eine Theme-Auswahl ist eine UI-Einstellung – halten Sie sie vom Ticket-Service getrennt. Wenden Sie das Theme an und aktualisieren Sie ein kleines Status-Label; verknüpfen Sie es nicht mit Speicher- oder Validierungscode.

```
private void cboTheme_SelectedIndexChanged(object sender, EventArgs e)
{
    string selectedTheme = cboTheme.Text;

    // Das ausgewählte Theme anwenden – diese UI-Einstellung getrennt
    // von der Logik des Ticket-Service halten.
    Application.Theme.Name = selectedTheme;
    lblCurrentTheme.Text = "Current theme: " + selectedTheme;
}
```

4. Checkpoint für Modul 7

- Können Sie erklären, was das Theme verändert?
- Können Sie die Geschäftslogik benennen, die sich *nicht* ändern sollte?
- Können Sie die aktuelle Seite am Navigationszustand erkennen?
- Können Sie erklären, warum Karten, Beschriftungen und Abstände die App leichter bedienbar machen?



Sie sind bereit für das Lab

Als Nächstes: Sehen Sie sich den Video-Walkthrough zur Modernisierung an, bestehen Sie den Wissenstest und gestalten und modernisieren Sie dann das Dashboard – wenden Sie ein integriertes Theme an, gruppieren Sie Kennzahlen in Karten, zeigen Sie den aktiven Navigationszustand und sorgen Sie dafür, dass jeder bestehende Arbeitsablauf weiter funktioniert.