

Immagini del tema, URL, colore degli SVG e l'Image Picker del designer

Guida della lezione · Guida al lab / esame



Guida della lezione

Sorgenti di immagine che non sono file distribuiti accanto alla pagina: immagini del tema caricate per nome dalla cache del tema sul client, URL relativi e assoluti, dati base64 incorporati nel tema, l'Image Selector di Visual Studio che scrive per te la stringa della sorgente e il suffisso ?color= che ricolora un SVG monocromatico adatto con il nome di un colore del tema. Che cosa significa per uno sviluppatore che conosce le basi di C# e ha già lavorato un po' con i controlli e il designer di Wisej.NET — e come affrontare questo modulo.



Focus di produzione

Sorgenti di immagine che non sono file distribuiti accanto alla pagina: immagini del tema indicate per nome e caricate dalla cache del tema sul client, URL relativi e assoluti, dati base64 incorporati nel tema, l'Image Selector di Visual Studio che scrive per te la stringa della sorgente e il suffisso ?color= che ricolora un SVG monocromatico adatto con il nome di un colore del tema.

1. Il problema in produzione

IconDesk ha ormai un pannello dei comandi e un image lab, e ogni immagine che vi compare è un file distribuito più un percorso che qualcuno ha scritto a mano. Regge finché l'applicazione non viene ritematizzata, distribuita sotto una virtual directory diversa oppure chiamata a mostrare lo stesso simbolo in un colore di avviso su una schermata e in uno neutro su un'altra. Questo modulo sposta l'origine di un'immagine fuori dalla cartella del progetto e dentro il tema, un URL o il designer, così che la stringa portata da un controllo continui a significare qualcosa dopo che la grafica, il tema o il server sono cambiati.

2. Concetti chiave

Concetto	Perché è importante
immagine del tema indicata per nome	Un nome semantico come <code>icon-print</code> che il tema attivo mappa su una grafica, così un tema diverso può fornire un'immagine diversa senza alcuna modifica in C#.
cache del tema sul client	Le immagini del tema viaggiano con il tema e sono già nel browser quando un controllo ne chiede una: questo rende il tema la casa giusta per un piccolo insieme di simboli di comando e quella sbagliata per migliaia di icone.
sorgente con URL relativo	Una stringa di sorgente immagine risolta rispetto all'applicazione: la forma più semplice per un logo distribuito insieme all'applicazione.
sorgente con URL assoluto	Un indirizzo esterno completo che trasforma l'immagine in una dipendenza da un altro sistema, con conseguenze su disponibilità, privacy, CORS e Content Security Policy.
immagine base64 incorporata	I byte dell'immagine scritti dentro la definizione del tema stessa, così la grafica non richiede una seconda richiesta, al prezzo di un tema più pesante.
Image Selector	Il selettore di Visual Studio in fase di progettazione che esplora le immagini del progetto, le immagini del tema e le risorse dei pacchetti di icone referenziati, poi serializza la stringa di sorgente scelta dentro <code>Designer.cs</code> .
suffisso <code>?color=</code>	Un suffisso su una sorgente immagine che chiede a Wisej.NET di ricolorare un SVG monocromatico compatibile con un colore in stile CSS o con il nome di un colore del tema.

3. Il filo della lezione

Quattro forme che un'immagine può assumere dentro un tema

Un tema di Wisej.NET non è fatto soltanto di colori e bordi; porta con sé anche definizioni di immagini che il framework e la tua applicazione possono usare entrambi. La documentazione del Theme Builder descrive quattro forme utili per quelle definizioni. Un **URL relativo** viene risolto rispetto all'applicazione. Un **URL assoluto** nomina un indirizzo esterno completo. I dati **base64 incorporati** mettono i byte dell'immagine dentro la definizione del tema, così non c'è nient'altro da recuperare. Un **immagine indicata per nome** punta a una definizione di immagine, e questo aggiunge un livello di indirezione: molte appearance possono fare riferimento allo stesso nome, e modificare la definizione una sola volta cambia ogni uso.

L'indirezione è la parte interessante per IconDesk. Quando la gallery chiede un nome invece di un file, la decisione su quale grafica quel nome indichi appartiene al tema, e il tema si può sostituire al momento della distribuzione o per singolo cliente. Quando la pagina chiede un URL, la decisione appartiene a chi ha scritto l'URL, e cambiare tema non cambia nulla. Il lab mette entrambi i comportamenti sulla stessa schermata, così la differenza non è una questione di fiducia.

Gli URL assoluti meritano una decisione deliberata, non un'abitudine. Un'immagine recuperata da un altro host è una dipendenza a runtime: può essere non disponibile, può richiedere un'autenticazione che il browser non ha, può essere bloccata da una Content Security Policy, e la sua politica di cache non è tua.

Un'icona che deve essere sempre presente è meglio distribuirla con l'applicazione o impacchettarla come risorsa, ed è la strada che prende il Modulo 5.

Il nome di un'immagine del tema è una chiave, non un percorso

Assegnare un'immagine del tema è la riga più breve del corso: imposta una proprietà `ImageSource` sul nome, e Wisej.NET la carica dalla cache del tema lato client. Nessuno apre un file sul server e nessun `System.Drawing.Image` viene creato, quindi questa è la seconda pipeline del Modulo 1, non la prima. La stringa è una chiave del tema. Se un altro tema mappa quella chiave su una grafica diversa, l'applicazione cambia aspetto senza una sola modifica in C#, ed è per questo che nomi semantici come `icon-print`, `icon-save` o `icon-warning` battono un percorso incollato.

Quel comportamento di precaricamento è anche una regola di dimensionamento. Le immagini del tema arrivano con il tema, quindi un tema si adatta a un insieme piccolo e stabile di simboli di comando a livello di framework. Le risorse dei pacchetti di icone si comportano al contrario: ogni icona viene richiesta singolarmente e poi messa in cache dal browser, il che regge cataloghi molto ampi ma fa costare una richiesta al primo uso di ciascuna icona. Sapere quale dei due stai usando ti dice che aspetto dovrebbe avere il pannello di rete, ed è proprio la prova che il Modulo 4 ti chiede di raccogliere.

```
// IconGallery: tre stringhe di sorgente, tre risoluzioni diverse.
buttonPrint.ImageSource = "icon-print";           // chiave del tema
logo.ImageSource        = "Images/company-logo.svg"; // URL relativo
avatar.ImageSource      = "https://cdn.example.com/users/42.jpg"; // assoluto

// Le scelte a runtime restano semantiche, mai costruite dal testo dell'utente.
buttonStatus.ImageSource = isDanger
    ? "icon-warning?color=error"
    : "icon-info?color=activeText";
```

Lascia che sia l'Image Selector a scrivere la stringa della sorgente

Il designer di Wisej.NET in Visual Studio offre un'esperienza di selezione delle immagini per le proprietà di sorgente. Esplora le immagini presenti nel progetto, le immagini definite dal tema corrente e, quando gli assembly dei pacchetti di icone sono referenziati, i pacchetti che scopre automaticamente. L'icon-pack explorer mostra un elenco o una griglia con anteprima, e un'icona monocromatica si può vedere in anteprima in un colore scelto prima di confermare. Quando ne scegli una, il designer serializza una sorgente concreta come un URL `resource.wx/...` dentro `Designer.cs`.

Non è soltanto un aiuto alla scrittura. Leggere la riga generata insegna la rappresentazione a runtime: scegliere l'icona di un pacchetto **non** copia l'SVG dentro la form. La form memorizza una stringa di sorgente che rimanda all'assembly del pacchetto, ed è per questo che cinque form che usano una stessa icona distribuiscono comunque un'unica copia della grafica. Il lab ti chiede di aprire `Designer.cs` e leggere quelle stringhe esattamente per questa ragione.

- Immagini del progetto: scegline una con il selettore, così nessun percorso viene scritto due volte.
- Immagini del tema: usa l'elenco dei temi e l'anteprima così da vedere la grafica del tema corrente.
- Risorse dei pacchetti di icone: cerca e guarda l'anteprima invece di memorizzare nomi di file.

Ricolorare un SVG monocromatico con ?color=

Wisej.NET può ricolorare un SVG monocromatico compatibile aggiungendo `?color=` alla sorgente immagine. Il valore può essere un colore in stile CSS come `red` o `#3078d4`, oppure il nome di un colore del tema come `error` o `activeText`. Un colore del tema è la scelta più solida perché continua ad avere senso quando il tema cambia: un grigio scuro fissato nel codice può sparire su un tema scuro, mentre un colore del tema semantico è pensato per adattarsi insieme a esso. Lo stesso suffisso funziona sul nome di un'immagine del tema e su una sorgente di risorsa `resource.wx`, perché si applica alla stringa della sorgente e non a un particolare meccanismo di archiviazione.

La ricolorazione non è universale, e il suo fallimento è silenzioso. La documentazione avverte che può fallire quando è l'SVG stesso a fissare al suo interno i valori di `fill` o `stroke`, perché quei valori prevalgono. Per un'icona destinata a essere colorata da Wisej.NET, rimuovi i colori fissati in conflitto dove è appropriato farlo; per un logo multicolore, lascialo stare. La gallery di IconDesk dovrebbe contenere uno di entrambi i casi, così che la nota di lab possa dire quale grafica ha accettato il suffisso e quale lo ha ignorato. Mantieni semantiche le stringhe di sorgente usate a runtime, e non costruire mai un percorso di risorsa a partire da testo non fidato dell'utente.



Dai prima un nome al meccanismo

Prima di assegnare qualsiasi immagine, dichiara quale meccanismo stai usando: un oggetto `System.Drawing.Image` che il server disegna come PNG, una stringa di sorgente immagine che il browser risolve, un'immagine del tema indicata per nome, una risorsa incorporata `resource.wx` o un font di icone disegnato come HTML. Assegna una sola proprietà immagine per controllo, così che le altre non possano azzerarla di nascosto, mantieni la stringa della sorgente significativa invece di un percorso magico incollato, preferisci l'SVG per le icone di interfaccia così che restino nitide a ogni dimensione e scala, e dai a ogni icona che porta un significato un'alternativa testuale.



Obiettivi di apprendimento

Alla fine di questo modulo sarai in grado di:

- ✓ Usare URL relativi, URL assoluti, dati base64 incorporati e immagini indicate per nome in un tema di Wisej.NET.
- ✓ Assegnare direttamente a `ImageSource` i nomi delle immagini del tema.
- ✓ Usare il selettore di immagini di Visual Studio in fase di progettazione per scegliere asset locali, del tema e dei pacchetti di icone.
- ✓ Ricolorare le icone SVG adatte con la sintassi `?color=` di Wisej.NET e i nomi dei colori del tema.

Guida al lab / esame

Che cosa costruirai nel lab pratico, l'approccio suggerito e i deliverable richiesti.



Lab pratico

Riempi una pagina IconGallery in IconDesk usando l'Image Selector del designer invece di scrivere a mano le stringhe di sorgente. Assegna immagini a cinque controlli: un'immagine del tema indicata per nome, un SVG del progetto referenziato tramite URL relativo, un URL assoluto e almeno due asset di un pacchetto di icone se un pacchetto è installato. Ricolora un SVG monocromatico con il suffisso `?color=` e il nome di un colore del tema, apri il Designer.cs generato e leggi le stringhe di sorgente immagine prodotte dal picker, poi passa l'applicazione da un tema Wisej a un altro e registra quali asset sono cambiati e quali sono rimasti esattamente uguali.

Punto di partenza

Apri la soluzione IconDesk dove l'ha lasciata il Modulo 2: il pannello dei comandi del Modulo 1 e la pagina ImageLab con la sua griglia di modalità di ridimensionamento, la sua PictureBox con `LoadAsync` e il suo SVG nitido. Aggiungi una nuova pagina chiamata `IconGallery` con cinque controlli vuoti e un selettore di tema, così che ogni immagine che vi compare sia assegnata durante questo lab e nulla sia ereditato da un modulo precedente.

Approccio suggerito

- ✓ Colloca prima i cinque controlli della gallery e dai a ciascuno una label che nomina il meccanismo che intendi usare, prima di assegnare qualsiasi immagine.
- ✓ Assegna ogni immagine attraverso l'Image Selector del designer: l'elenco dei temi per l'immagine del tema indicata per nome, l'albero del progetto per l'SVG relativo, il campo dell'indirizzo per l'URL assoluto e l'icon-pack explorer per gli asset dei pacchetti.
- ✓ Aggiungi il suffisso `?color=` con il nome di un colore del tema a un SVG monocromatico, e provalo anche su un asset multicolore così da vedere il suffisso venire ignorato.
- ✓ Apri `Designer.cs`, copia nella nota di lab le righe di sorgente immagine generate e segna ciascuna come chiave del tema, URL relativo, URL assoluto o sorgente di risorsa.
- ✓ Passa l'applicazione da un tema Wisej a un altro, cattura la gallery in entrambi e registra quali controlli hanno seguito il tema e quali no.

Deliverable richiesti

- ✓ Pagina IconGallery con cinque controlli le cui immagini sono state assegnate attraverso l'Image Selector del designer
- ✓ Tra di esse un'immagine del tema indicata per nome, un SVG relativo del progetto, un URL assoluto e almeno due asset di un pacchetto di icone
- ✓ SVG monocromatico ricolorato attraverso il suffisso `?color=` con il nome di un colore del tema
- ✓ Estratto di Designer.cs che mostra le stringhe di sorgente immagine generate dal picker
- ✓ Nota sul cambio di tema che registra quali asset hanno seguito il tema e quali sono rimasti fissi

Domande di autovalutazione

- ✓ La gallery si disegna correttamente sulla tua macchina e non mostra nulla per l'URL assoluto su quella di un collega. Quali dei cinque meccanismi potrebbero produrre quella differenza, e che cosa controlleresti per primo?
- ✓ Due delle tue icone si rifiutano di cambiare colore quando aggiungi il suffisso. Che cosa c'è dentro quei file, e quando modificarli è la risposta sbagliata?
- ✓ Un designer chiede lo stesso simbolo nella gallery, in una toolbar e in una cella di griglia. Tra la chiave del tema, l'URL relativo e la risorsa del pacchetto, quale costa meno cambiare in futuro, e perché?



Dai prima un nome al meccanismo

Prima di assegnare qualsiasi immagine, dichiara quale meccanismo stai usando: un oggetto `System.Drawing.Image` che il server disegna come PNG, una stringa di sorgente immagine che il browser risolve, un'immagine del tema indicata per nome, una risorsa incorporata `resource.wx` o un font di icone disegnato come HTML. Assegna una sola proprietà immagine per controllo, così che le altre non possano azzerarla di nascosto, mantieni la stringa della sorgente significativa invece di un percorso magico incollato, preferisci l'SVG per le icone di interfaccia così che restino nitide a ogni dimensione e scala, e dai a ogni icona che porta un significato un'alternativa testuale.