Knowlbo®/Component

TimeView for Web

タイムビュー for Web

リリース ***Version 1***

Knowlbo

2024 年 5 月 1 日

目 次

第1章　はじめに

Knowlbo(R)/Web シリーズ タイム ビュー for Web Version1 をお求めいただき誠にありがとうございます。

タイム ビューは、実用的な機能と豊富な表現力でさまざまな時間スケジュール情報をガント チャート形式で表示します。

1.1 マニュアルの内容

本マニュアルは、タイム ビューのソフトウェア使用許諾権、サポート情報、使用方法などの各種ドキュメント、およびサンプルコードなどから構成されています。また、最新の製品情報や修正版、FAQ などについては、弊社ホームページにて公開されます。

1.2 登録商標など

Knowlbo(R) はナルボの登録商標です。

Microsoft、Windows、Visual C#、Visual Studio は、米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

その他記載されている会社名、製品名は各社の登録商標、または商標です。

なお、本マニュアルの記述については、特に記載がない限り、C# および JavaScript 言語を対象としています。他の言語環境でご利用される場合は、お手数ですが対応の言語に置き換えてご利用下さい。

1.3 ソフトウェア使用許諾権

1.3.1 ソフトウェア使用許諾契約

お客様は、当ソフトウェア使用許諾の内容に同意してご利用いただけるものといたします。この内容に同意できない場合には、ご利用することはできません。

（「ソフトウェア使用許諾書」がある場合はこれも含めた内容が対象となります）

1.3.2 適用

以下の条項は、お客様がこの契約書とともに入手された株式会社ナルボ（以下「弊社」といいます）のソフトウエア製品（コンピュータプログラムとマニュアルその他の関連資料を含み、以下「本製品」といいます）に適用します。

1.3.3 使用の許諾

（1）お客様は「ユーザ登録」に記載された担当者の管理下においてのみ本製品を使用できます。

（2）お客様は本製品をコンピュータネットワーク上で複数の使用者により使用することはできません。ただし、マニュアルその他にネットワーク対応製品である旨が記載されている製品はこの限りではありません。

（3）お客様はいかなる事由によろうとも本製品を譲渡、販売、転貸することはできません。

（4）本製品のパッケージに同一の機能を有し、形態の異なるディスクが複数入っていた場合、お客様はそのうちの一種類のディスクのみをご使用いただけます。お客様が残りのディスクを他のコンピュータで使用したり、他者に譲渡、販売、転貸することはできません。また、お客様のご要望に応じて弊社が本製品と同一の機能を有し、形態の異なるディスクをお客様に提供した場合も同様とします。

（5）本製品のマニュアルその他にロイヤルティフリーの言語関連製品である旨が記載されている場合に限り、お客様が販売する製品に本製品のランタイムモジュールを組み込むことができます。ただし、この場合においても本製品の使用法やその他一切を公開することはできません。また、販売する製品には本製品の有効な著作権表示を明記しなければなりません。

1.3.4 保証の範囲

（1）弊社は本製品の仕様を予告なしに変更することがあります。

（2）本製品の品質および性能に関して発生する問題は、お客様の費用負担をもって処理することとします。

（3）弊社は、いかなる場合にも、お客様が本製品を使用または運用した結果に関して一切責任を負いません。

（4）本製品にお客様が目的を遂げることができないような重大な欠陥（磁気の消滅、破損など）があった場合、欠陥のない製品と交換するかまたはその製品代金相当額をもって弊社の保証限度とします。ただし、この保証は、お客様が本製品をご購入されてから 30 日以内に限らせていただきます。

1.3.5 テクニカルサポートサービス

弊社は、ユーザ登録によりご登録をいただいたお客様に対してのみ、テクニカルサポートサービスをご提供するものとします。但し、本製品（バージョン）の販売が終了して、１年以上が経過した場合にはその限りではありません。

1.3.6 契約期間

（1）本契約はお客様に製品を納入した日より発効します。また、納入日は証票により確認するものとします。

（2）お客様が本契約の条項のいずれかに違反した場合、本契約は解除されます。またこの場合を含め、製品の代金は返還いたしません。

1.4 ユーザ登録について

弊社では、よりよい製品とサービスをお届けするために、お客様の情報を登録させていただきます。お手数をおかけいたしますが、弊社ホームページまたは「ユーザ登録 FAX 用紙」を印刷のうえ、必要事項をご記入のうえファックスにてお送りください。

ご登録いただきましたお客様には、製品やサービスについてのインフォメーションをお届けいたします。ご住所や電話番号など、ご登録内容に変更が生じた場合は弊社カスタマ･コミュニケーションズまでお知らせください。

ご登録のない場合、バージョンアップサービスやテクニカルサポートサービスをご提供できないことがあります。

1.4.1 カスタマ・コミュニケーションズのご案内

ユーザ登録や本製品のご使用中に、README や ヘルプ、FAQ 情報で解決できない問題が発生した場合は、当社カスタマ・コミュニケーションズまでお問い合わせください。

1.4.2 カスタマ・コミュニケーションズ

電子メールアドレス

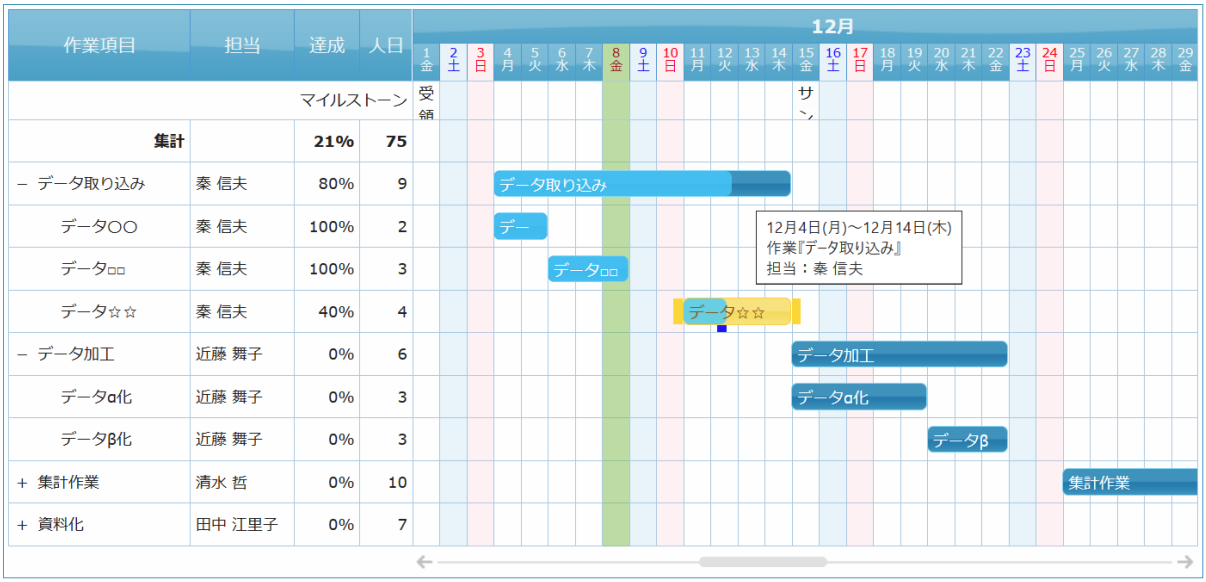
knocx@knowlbo.co.jp

第2章　機能紹介

2.1 コントロールの外観

タイムビュー コントロールはガントチャート形式の表示構造をしています。

管理する情報は作業や人のプロジェクト管理に限らず、施設予約スケジュール、結婚式などイベントのスケジュール表など様々な用途に利用が可能です。



縦方向に「アイテム」という作業項目のリストがあり、横方向に「タイムルーラー」という時間軸があり、アイテムとタイムルーラーで格子となっている場所に作業の期間やスケジュールといったものを表す「ピース」というバーを表示します。

2.2 UI テーマ対応

2.2.1 jQuery UI テーマ

タイムビュー コントロールは、jQuery UI のテーマ に対応しています。テーマファイルの入れ替えによって簡単にビジュアルを変更できます。

Pepper Grinder （コショウ挽き）

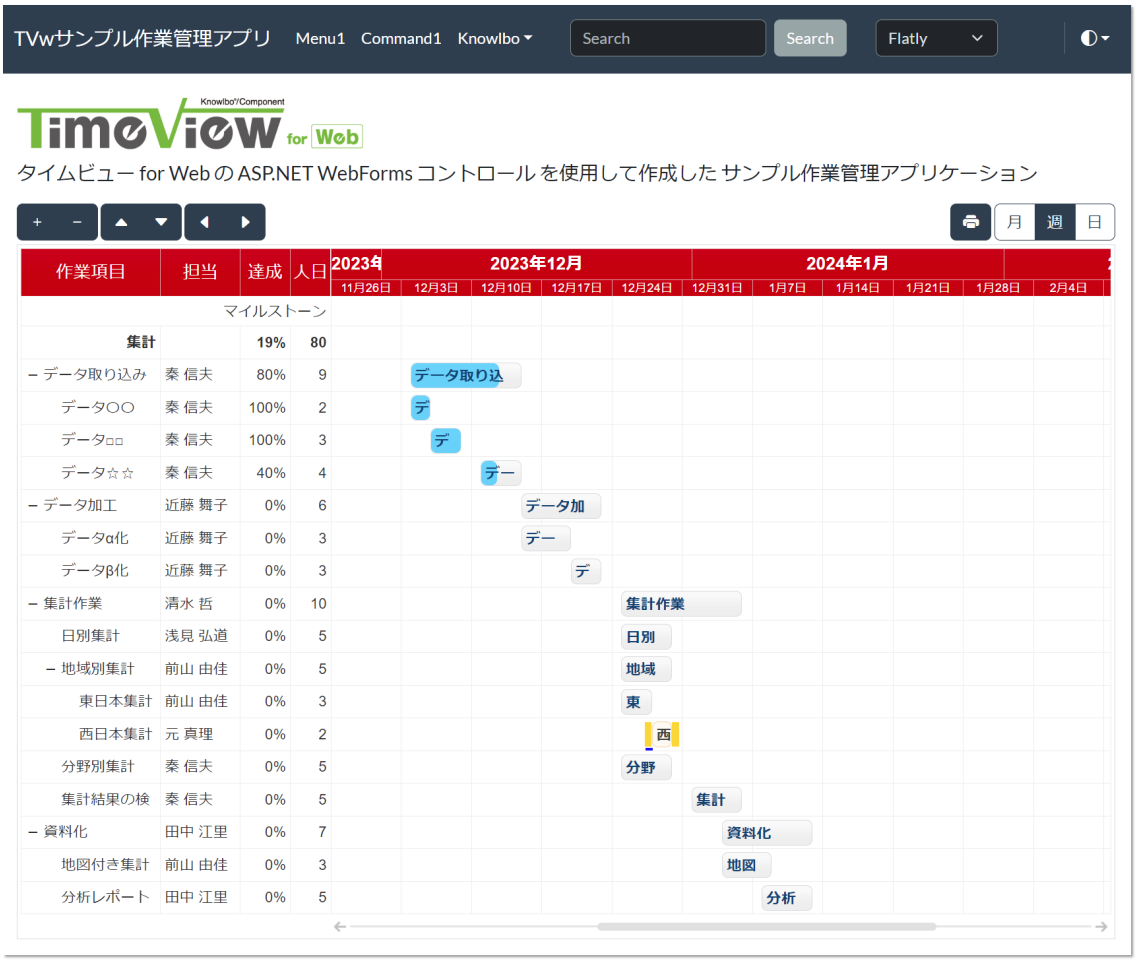
Le Frog （カエル）

Eggplant （茄子）

2.2.2 Bootstrap テーマ

また、 Bootswatch といった Bootstrap のテーマとも親和性があり、自然にフォントや表示サイズなどのスタイルが連動して対応します。

Blitzer


TVwサンプル作業管理アプリ
Menu1
Command1
Knowlbo
Search
Search
Flatly
TimeView for Web
タイムビュー for Web の ASP.NET WebForms コントロールを使用して作成した サンプル作業管理アプリケーション
月
週
日
作業項目
担当
達成
人日
2023年
2023年12月
2024年1月
11月26日
12月3日
12月10日
12月17日
12月24日
12月31日
1月7日
1月14日
1月21日
1月28日
2月4日
マイルストーン
集計 19% 80
– データ取り込み 秦 信夫 80% 9
データ〇〇 秦 信夫 100% 2
データ□□ 秦 信夫 100% 3
データ☆☆ 秦 信夫 40% 4
– データ加工 近藤 舞子 0% 6
データα化 近藤 舞子 0% 3
データβ化 近藤 舞子 0% 3
– 集計作業 清水 哲 0% 10
日別集計 浅見 弘道 0% 5
– 地域別集計 前山 由佳 0% 5
東日本集計 前山 由佳 0% 3
西日本集計 元 真理 0% 2
分野別集計 秦 信夫 0% 5
集計結果の検 秦 信夫 0% 5
– 資料化 田中 江里 0% 7
地図付き集計 前山 由佳 0% 3
分析レポート 田中 江里 0% 5
データ取り込
データ加
集計作業
日別
地域
東
西
分野
集計
資料化
地図
分析

Sketch

2.3 タイムルーラー

- 大区分、中区分、小区分の最大３段で目盛り表示できます。
- 大区分と中区分の目盛りは非表示にすることも可能です。
- 各区分毎に、目盛りの単位（年 / 月 / 週 / 日）と表記を設定できます。

大区分：月、中区分：週、小区分：日

大区分：非表示、中区分：月、小区分：週

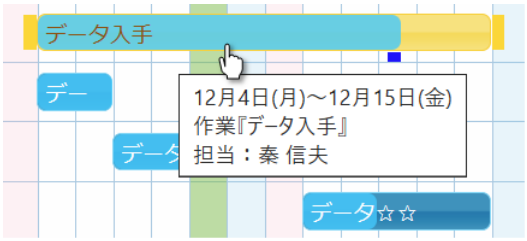
大区分：非表示、中区分：年、小区分：月

2.4 特別時間帯

特別時間帯を使用して、一部の日付を任意のカラーで強調表示できます。例えば、会社のイベントやプロジェクトのマイルストーンとなる日をわかりやすく表せます。特別時間帯の設定は、いくつもの日付に設定できます。また、任意のテキストを設定することができ、設定した日付にマウスを移動すると表示させることができます。

2.5 ピース、達成率

- 「ピース」という単位で１つの作業期間を入力します。
- 指定した開始日と終了日を繋いだバー形式で表示します。
- 同じアイテム上に複数のピースを入力して、離れた作業期間も表現できます。
- ピースに対して達成率をパーセンテージで設定・表示できます。
- ピースごとに任意のテキストや内部的なキー値などを設定できます。
- ピースに接敵したテキストはツールチップとして表示することができます。

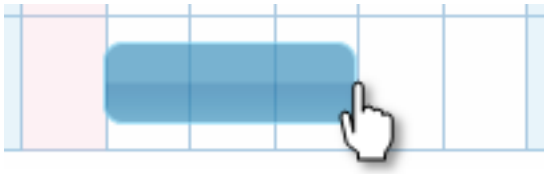


2.6 ユーザーインターフェース

スケジュール データを表示するだけでなく、マウス操作によって直接入力する機能も備えています。また、マウスによって行った操作に対応するイベント通知も備えているため、プログラムによって操作を受理、拒否を制御できます。

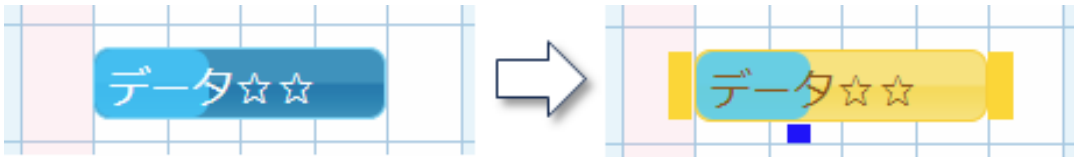
2.6.1 ピース追加

アイテム上で希望の開始日から終了日へマウスをドラッグすることで追加できます。



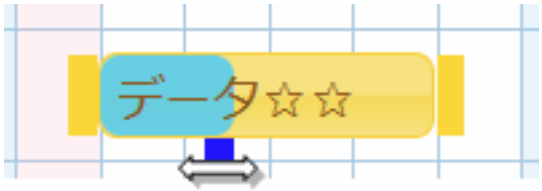
2.6.2 ピース選択

ピースをクリックして選択できます。ピース選択機能の有効・無効も設定できます。



2.6.3 達成率変更

ピースを選択すると表示される達成率移動ポイントをドラッグして変更できます。



2.6.4 ピース期間変更

ピースを選択すると両端に表示される期間変更ポイントをドラッグして変更できます。また、ピースの真ん中付近を押下してドラッグすることでピース全体を移動できます。

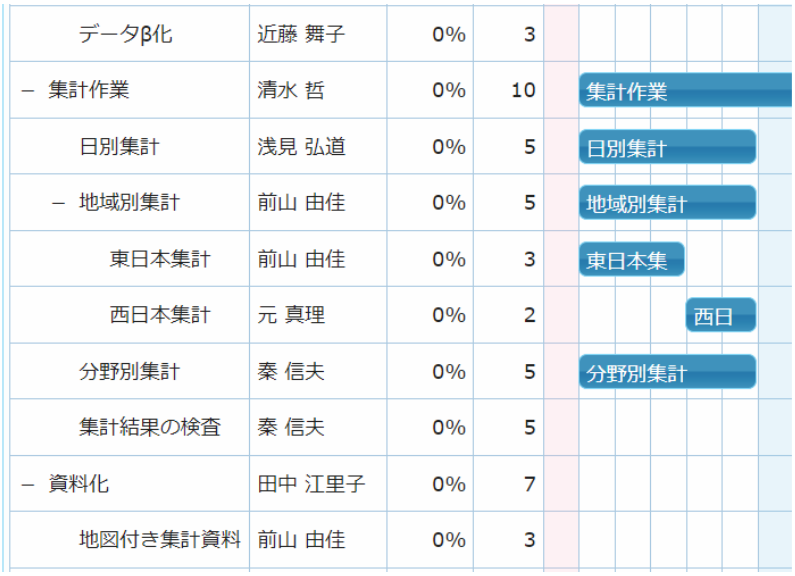
2.7 複数列とセル

- 列を複数作成して表示できます。
- 列の追加によってより多くの情報を表示させることが可能になります。
- 列ごとに任意の表示テキストと列幅を設定できます。

2.8 アイテムの階層化

- アイテムは階層レベルを設定できます。
- レベルに応じたインデント（字下げ）表示によって階層を表示します。
- 下位レベルのアイテムを持つアイテムには［−］または［+］記号が表示されます。これをクリックすることで下位レベルのアイテムを非表示にしたり、再表示できます。

階層を展開

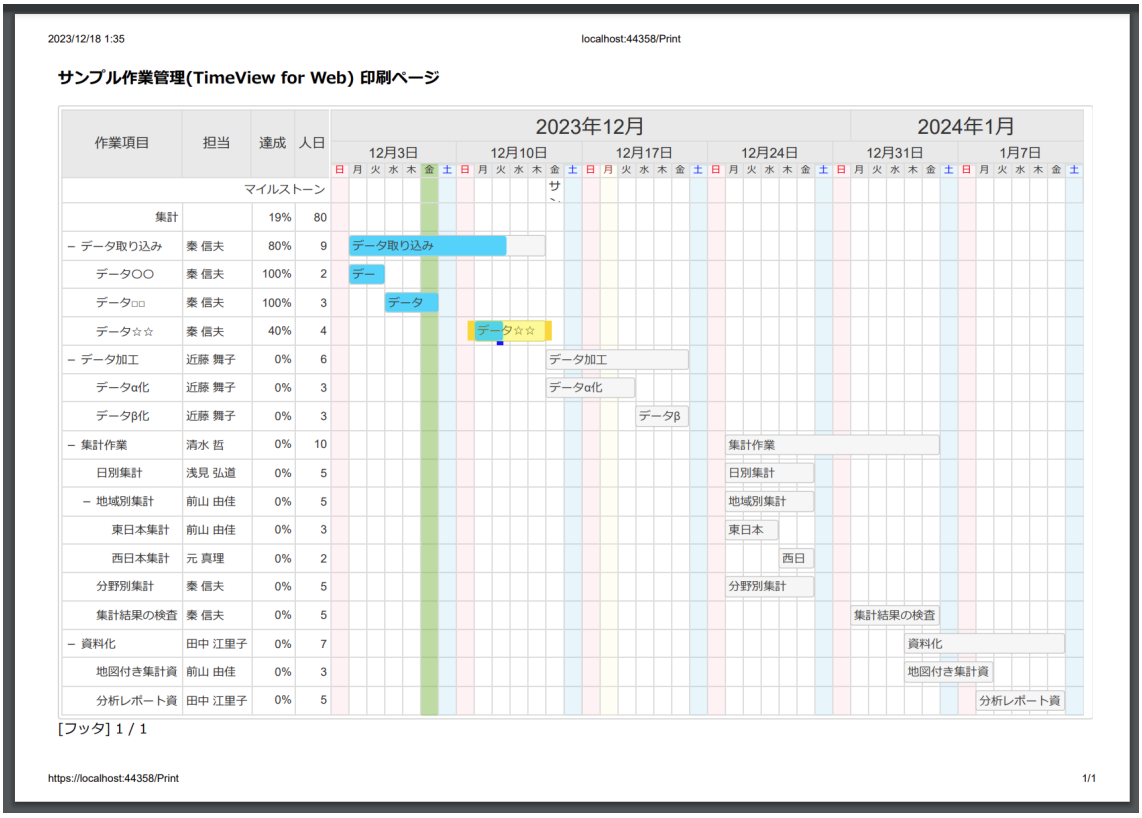


階層を閉じる

2.9 印刷

専用の印刷向けの表示機能を使用して、タイム ビュー コントロールのみを複数ページに印刷することができます。1 ページあたりのアイテム行数および時間範囲の指定や、ヘッダやフッタ領域に出力するテキストの指定が可能です。

印刷出力



2.10 2 つのプログラミング インターフェース

タイムビュー for Web には、ASP.NET WebForms のサーバーサイド開発に適した ASP.NET サーバー コントロール、および クライアントサイドのみで完結できる jQuery UI コントロールのプログラミング インターフェースを提供しています。

2.10.1 ASP.NET サーバーコントロール

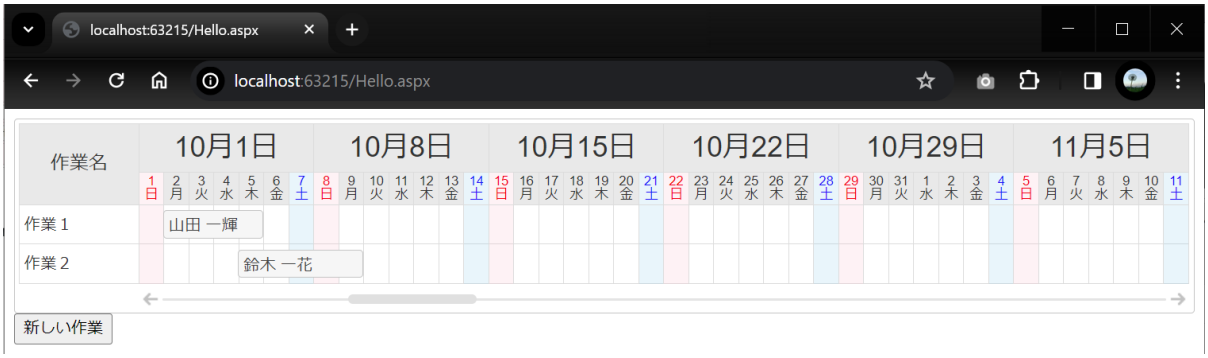
ASP.NET WebForms プロジェクトでの開発に用いることができます。

Hello.aspx.cs | Hello.aspx

3_Hello.aspx | Hello | Page_Load(object sender, EventArgs e)

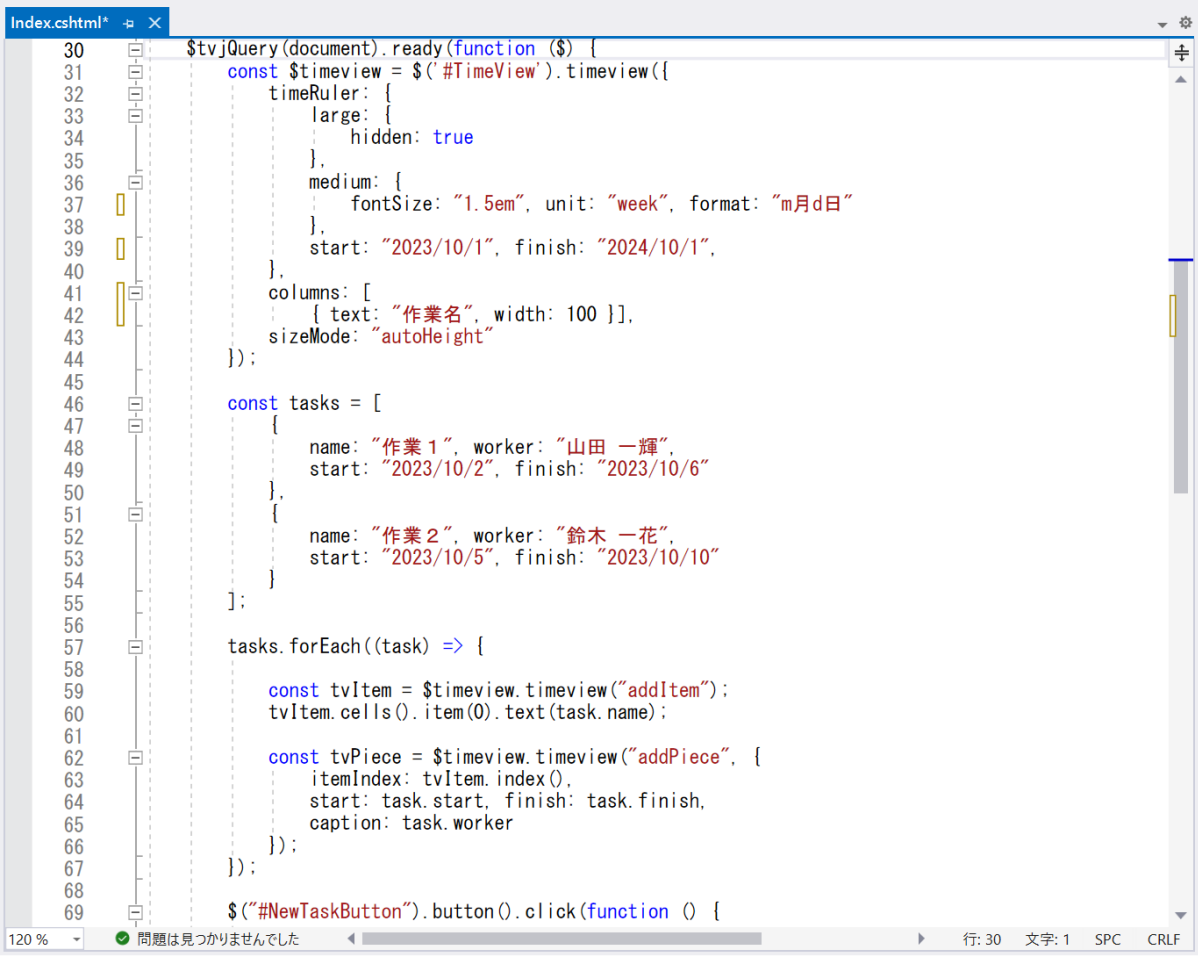
```
1  using KnowIbo.Component.Web.TimeView;
2  using System;
3
4  public partial class Hello : System.Web.UI.Page
5  {
6
7      protected void Page_Load(object sender, EventArgs e)
8      {
9          if (!IsPostBack)
10         {
11             KnTView1.Columns.Add(new Column()
12             {
13                 Text = "作業名",
14                 Width = 100
15             });
16
17             KnTView1.Ruler.Large.Hidden = true;
18             KnTView1.Ruler.Medium.Format = "m月d日";
19             KnTView1.Ruler.Medium.FontSize = "1.5em";
20             KnTView1.Ruler.VisibleStartTime = new DateTime(2023, 10, 1);
21
22             var tasks = new[] {
23                 new { Name = "作業１", Worker = "山田 一輝",
24                       Start = new DateTime(2023, 10, 2),
25                       Finish = new DateTime(2023, 10, 6) },
26                 new { Name = "作業２", Worker = "鈴木 一花",
27                     Start = new DateTime(2023, 10, 5),
28                     Finish = new DateTime(2023, 10, 10) }
29             };
30
31             foreach (var task in tasks)
32             {
33                 var item = new Item(task.Name);
34                 item.Pieces.Add(new Piece(task.Start, task.Finish, task.Worker));
35
36                 KnTView1.Items.Add(item);
37             }
38         }
39     }
40
41     protected void Button1_Click(object sender, EventArgs e)
42     {
```

120 % | 問題は見つかりませんでした | 行: 11 | 文字: 24 | SPC | CRLF



2.10.2 jQuery UI コントロール

ASP.NET MVC や ASP.NET Core MVC のみならず、Django や Laravel などさまざまな Web アプリケーション開発環境のクライアント サイド コンポーネントとして利用することができます。



第3章　動作環境

3.1 開発環境

- **OS(基本ソフトウェア)**
  - Windows 10 / 11 （日本語版）
- **ライブラリ**
  - .NET Framework 4.6.2 以降
  - jQuery 1.12.4 *1
  - jQuery UI 1.13.2 *1

    *1 TimeView jQuery コントロールを用いる場合に必要です。TimeView ASP.NET WebForms コントロールでは外部へ影響しない形で内蔵しているため不要です。
- **言語**
  - C#
  - JavaScript, HTML, CSS
- **IDE**
  - **Visual Studio 2022 （日本語版）*2 *3 *4**

    *2 動作確認およびサンプル提供の言語は C#となります。

    *3 .NET サンプルのソリューションは Visual Studio 2022 で作成しています。

    *4 TimeView jQuery コントロールは、一般的な jQuery コントロールを用いる開発と同様に VS Code など任意のテキストエディタ上で開発することも可能です。

3.2 実行環境

- **TimeView ASP.NET WebForms コントロール**
  - **サーバー**
    - Windows 10 / 11 （日本語版）
    - Windows Server 2019 （日本語版）
    - Azure App Service (Windows)

  - **クライアント**

    * ブラウザ (Google Chrome, Microsoft Edge, Safari, Firefox) の各最新版

- **TimeView jQuery コントロール**

  - **サーバー**

    * 不問（クライアントのブラウザ上でのみ実行されます）

  - **クライアント**

    * ブラウザ (Google Chrome, Microsoft Edge, Safari, Firefox) の各最新版

第4章　セットアップ

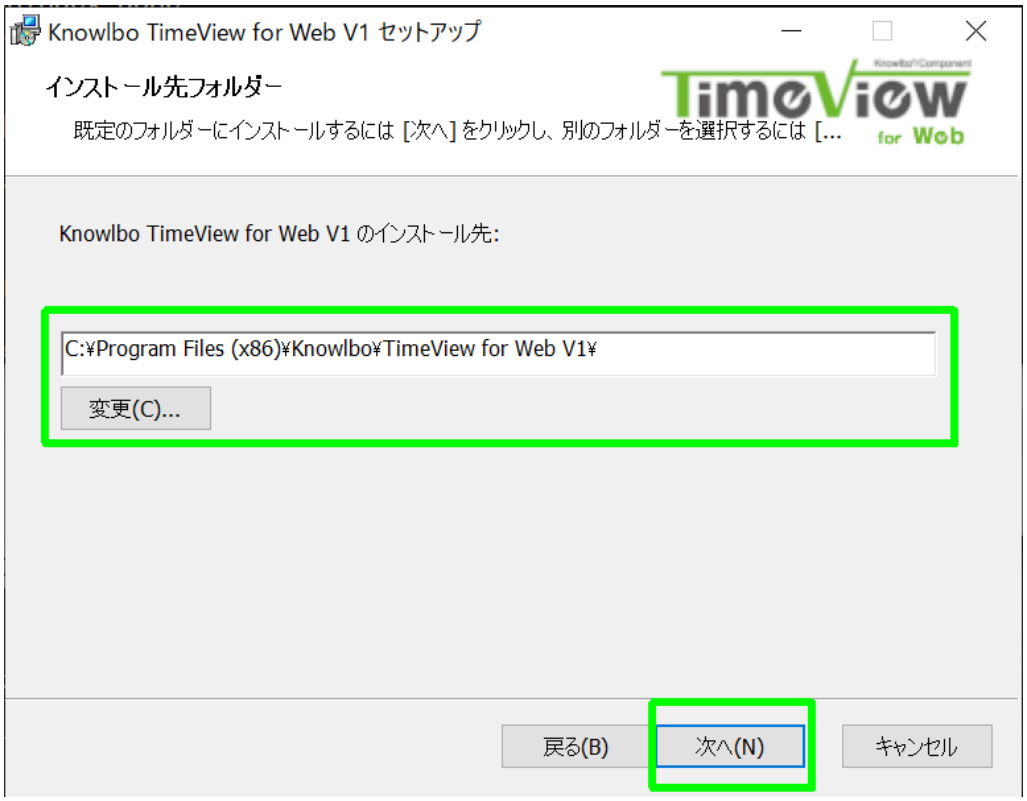
4.1 インストール

1. 開発に用いるコンピュータの任意フォルダへ タイムビュー for Web のインストーラ（TimeViewForWebInstaller-V1.msi）を配置します。

2. インストーラをダブルクリックして実行します。
3. 「ようこそ」画面の「次へ」をクリックします。

4. 使用許諾契約の内容をご確認の上、ご同意いただけます際は「使用許諾契約書に同意します」をクリックしてチェックを入れ、「次へ」をクリックします。

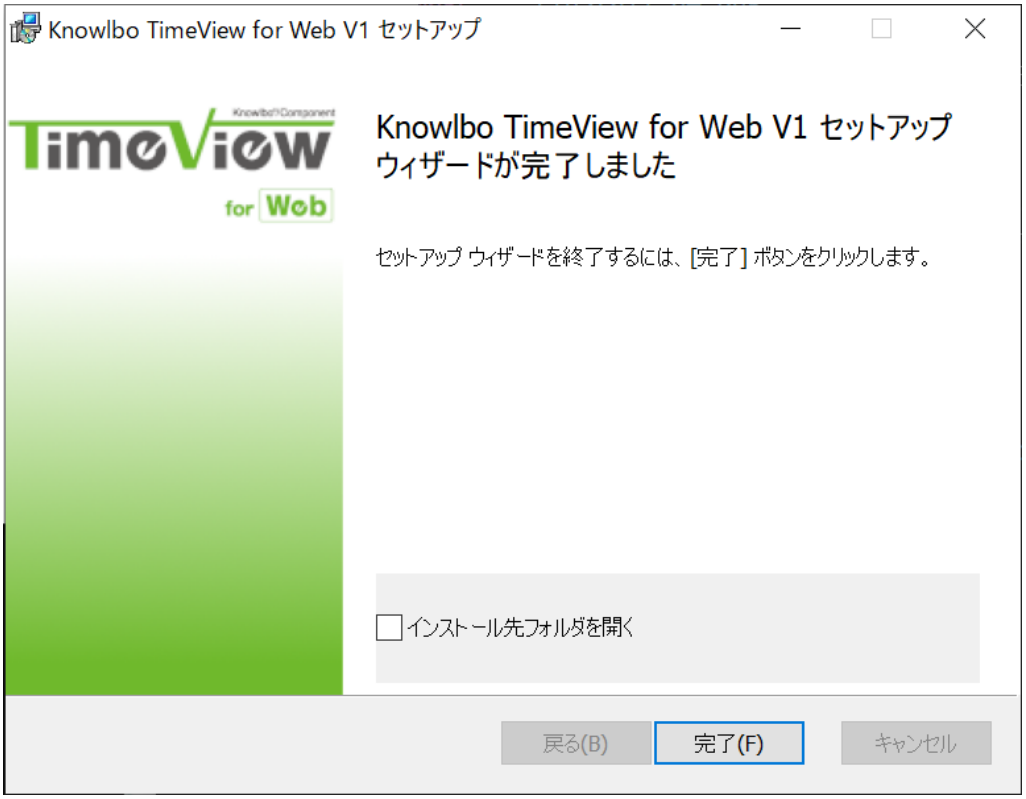
5. 「インストール先フォルダ」画面にて、必要に応じてインストール先フォルダを変更の上、「次へ」をクリックします。



6. 「インストール準備完了」画面にて、インストールを実行してよろしければ「インストール」をクリックします。

7. 「このアプリがデバイスに変更を加えることを許可しますか？」のメッセージが表示された場合は、「はい」をクリックします。

8. 「完了」画面が表示されたら、インストール成功です。
   「完了」をクリックしてインストーラを終了します。
   (先に「インストール先フォルダを開く」をチェックしておくとインストーラ終了時にインストール先フォルダを表示します)



4.2 アンインストール

本製品をアンインストール（削除）する際の手順は、以下の通りです。

1. コントロールパネルの「アプリと機能」を開きます。

2. 「Knowlbo TimeView for Web V1」を選択して、「アンインストール」をクリックします。

3. アンインストールの実行してもよい場合は「アンインストール」をクリックします。

4. タイムビュー for Web のアンインストールが開始します。
5. 「アプリと機能」の一覧から「Knowlbo TimeView for Web V1」が消えてアンインストール（削除）が完了します。

第5章　使用開始

5.1 TimeView ASP.NET WebForms コントロールを用いた開発

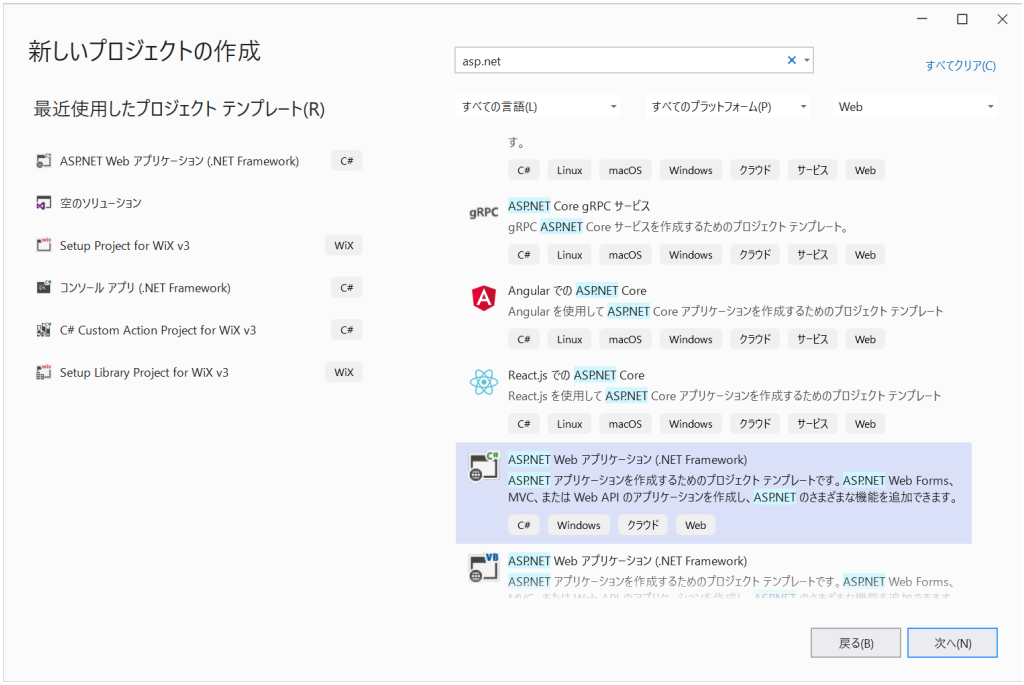
TimeView for Web の ASP.NET WebForms コントロールを使用した Web アプリケーションの作成をはじめる手順を示します。

この例では Visual Studio Community 2022 を使用しています。

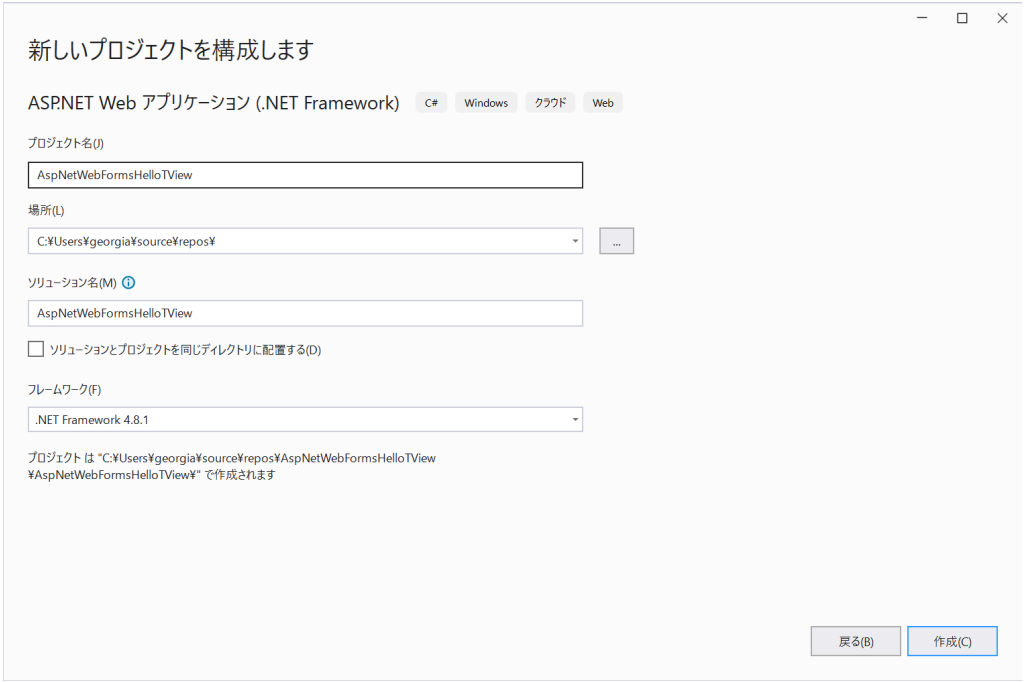
目次

5.1.1 新しい ASP.NET WebForms アプリケーション プロジェクトを作成する

1. 新しいプロジェクトとして「ASP.NET Web アプリケーション (.NET Framework) C#」を選択します。



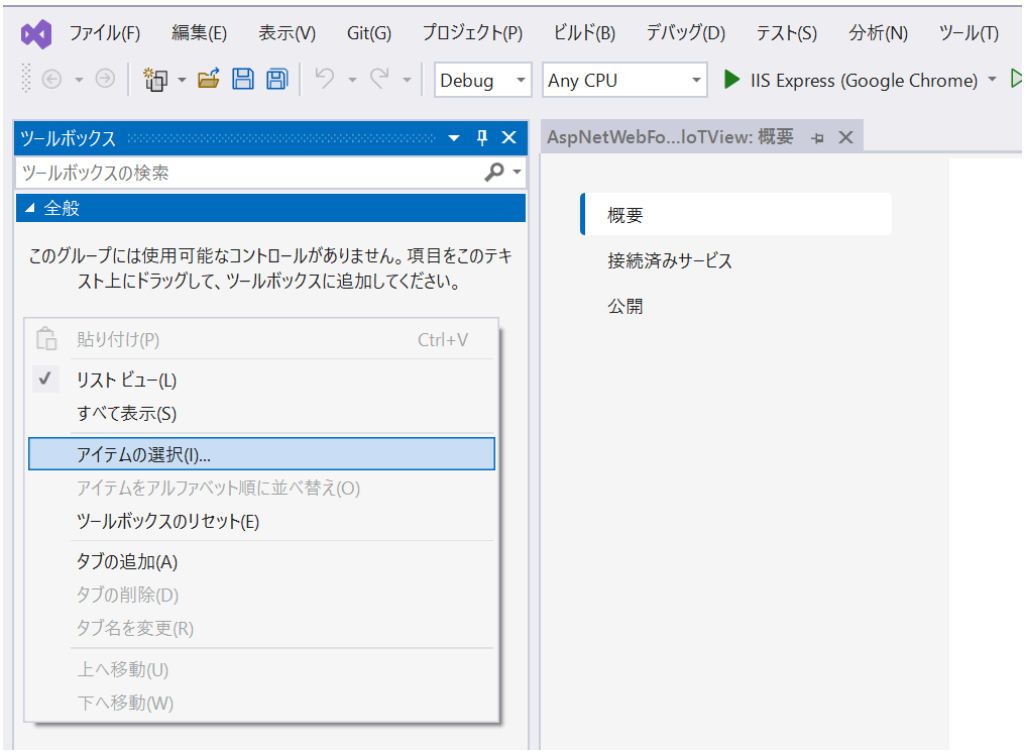
2. フレームワークは .NET Framework 4.6.2 以上を選択します。



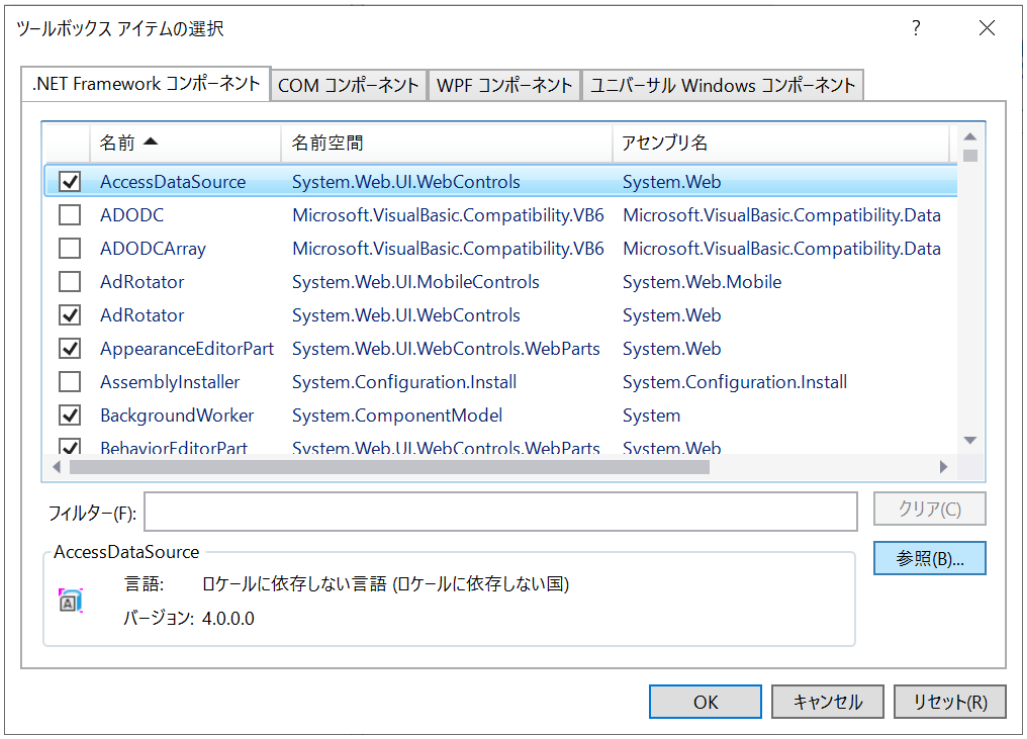
3. Web Forms を選択して作成します〇〇を押します。

5.1.2 ツールボックス に タイムビュー を追加する

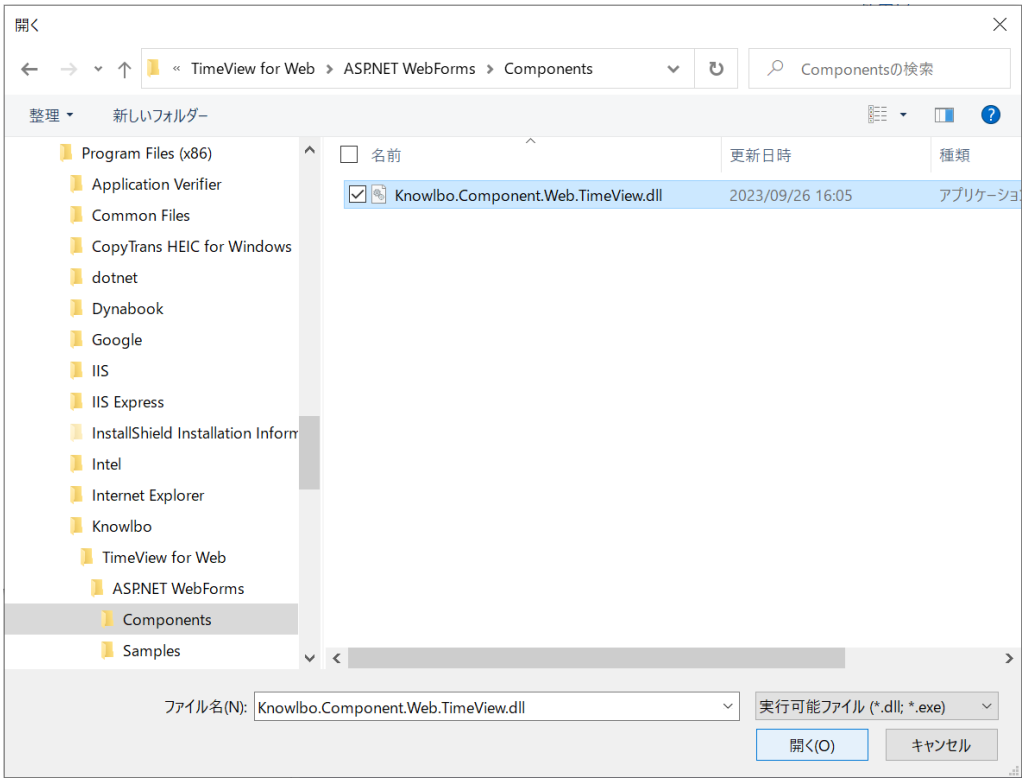
1. ツールボックスの「全般」内で右クリックして「アイテムの選択...」を選択します。



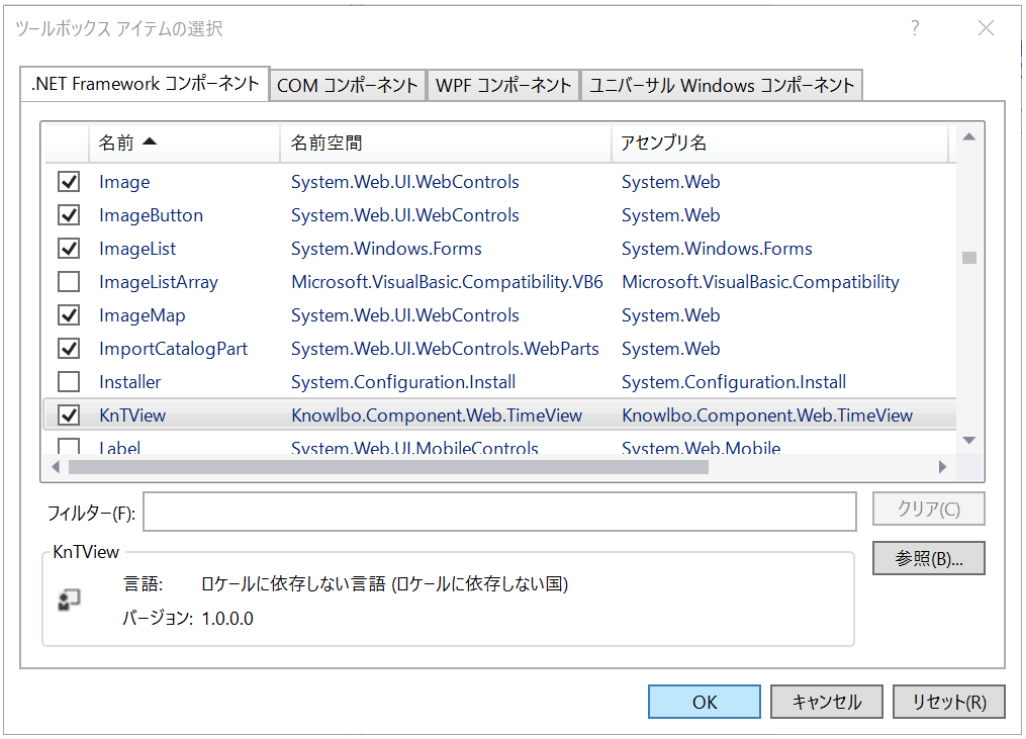
2. 「参照...」ボタンをクリックします。



3. タイムビュー for Web のインストール先フォルダ（既定では C:\Program Files (x86)\Knowlbo\TimeView for Web V1）内の ASP.NET WebForms\Components フォルダまで移動し、Knowlbo.Component.Web.TimeView.dll を選択して「開く」ボタンをクリックします。

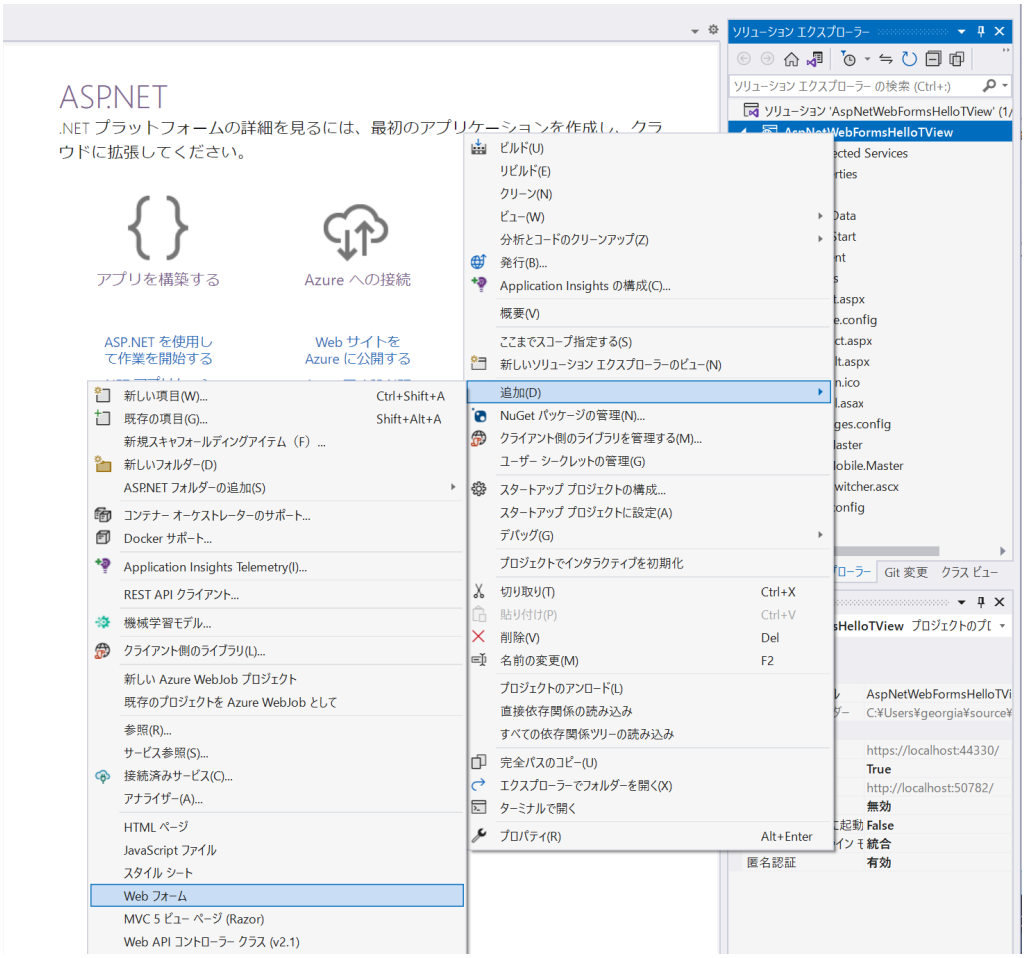


4. KnTView が追加されていることを確認して、「OK」ボタンをクリックします。



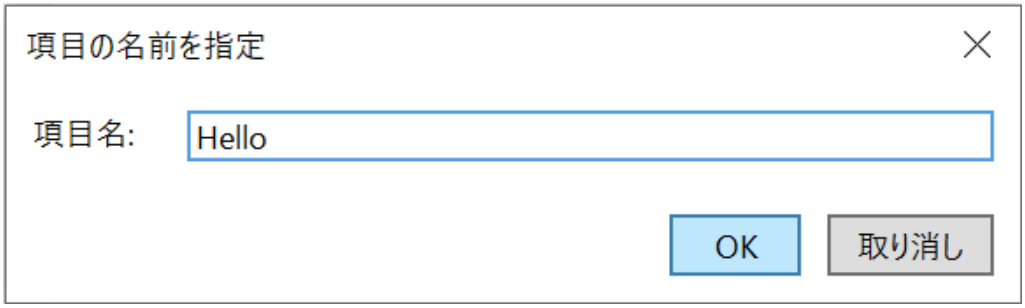
5.1.3 Web フォーム を追加する

1. プロジェクトを右クリック → 「追加」 → 「Web フォーム」 を選択します。

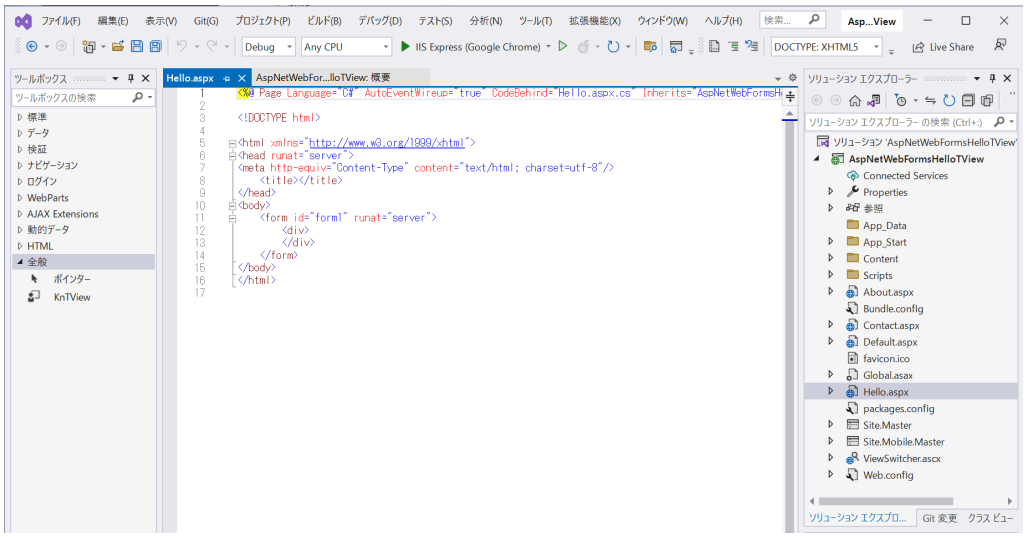


（「マスター ページを含む Web フォーム」を選択した場合も以下同様に作業できます）

2. ファイル名を任意に入力して「OK」ボタンをクリックします。

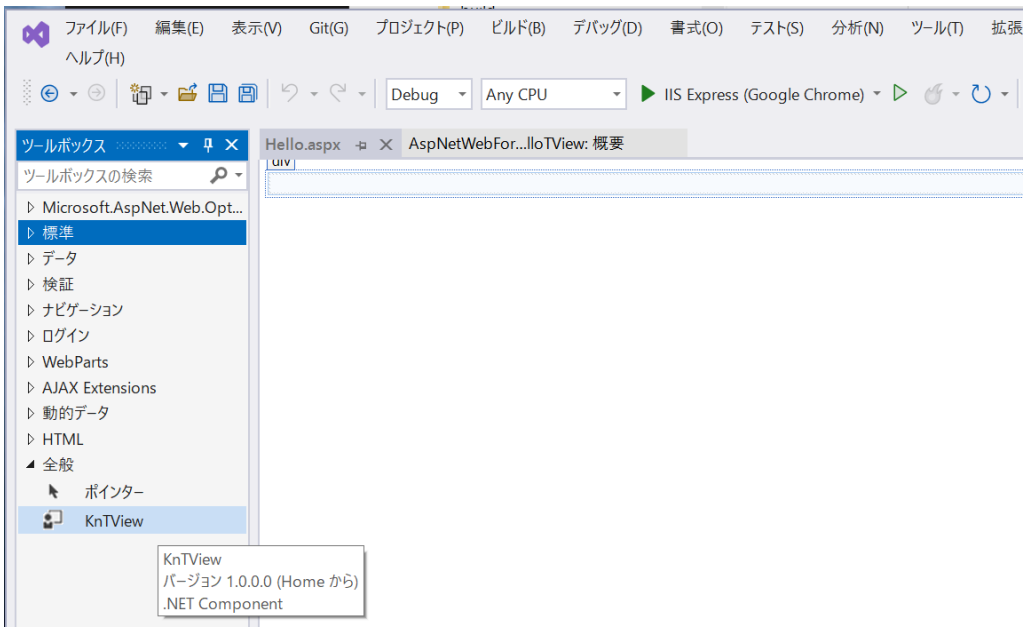


3. 作成直後でタイムビューを追加する前の Web フォームのソースは以下のようになっています。

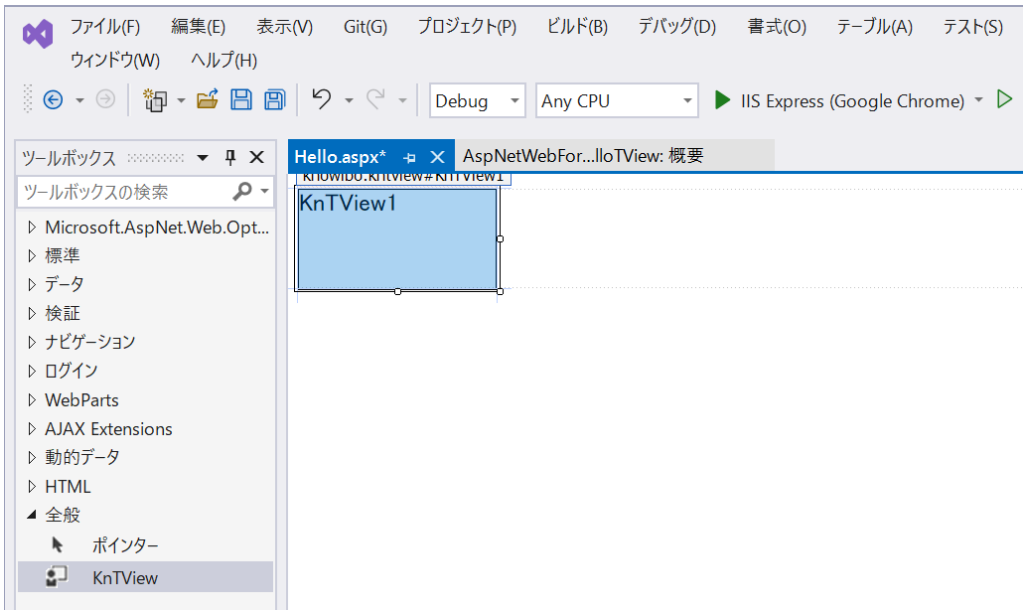


5.1.4 Web フォーム に タイムビュー を追加する

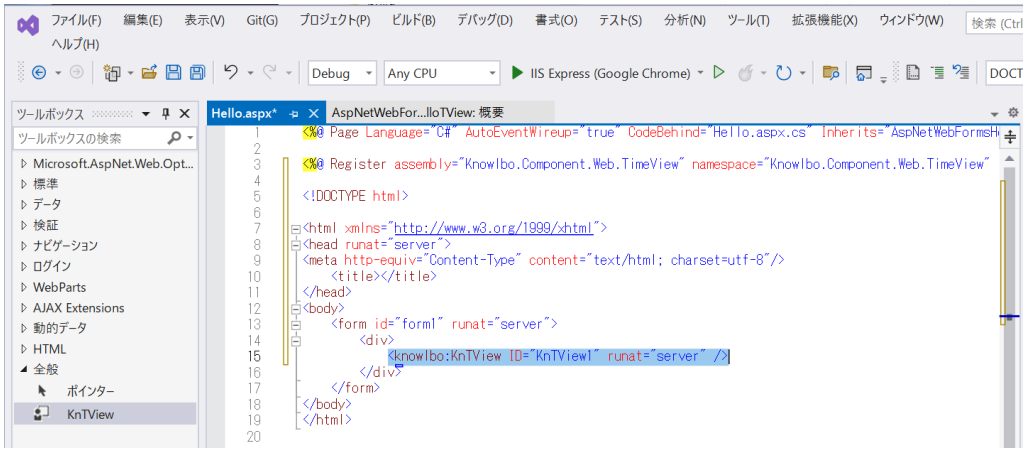
1. 「ツールボックス」の「全般」にある「KnTView (タイムビュー)」を Web フォームの追加したい場所へドラッグアンドドロップします。



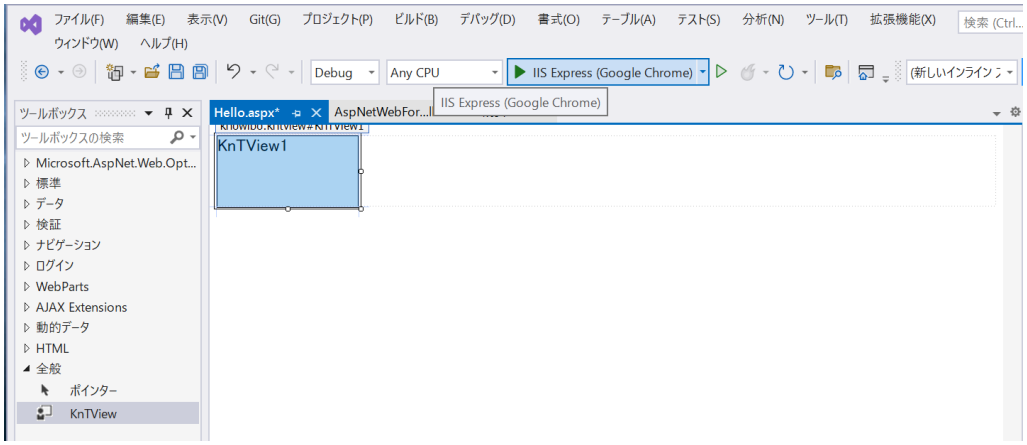
2. タイムビュー が Web フォームに追加されます。



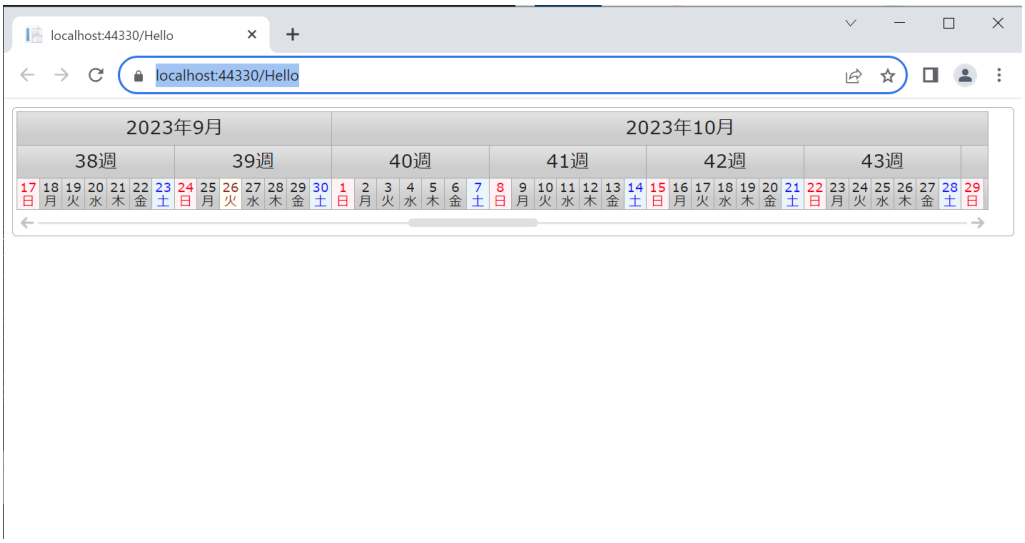
3. Web フォームを「ソース」タブへ切り替えますと、@Register ディレクティブ や タイムビュー コントロールのタグが追加されていることが確認できます。



4. F5 キーを押してビルドとデバッグの開始を行います。

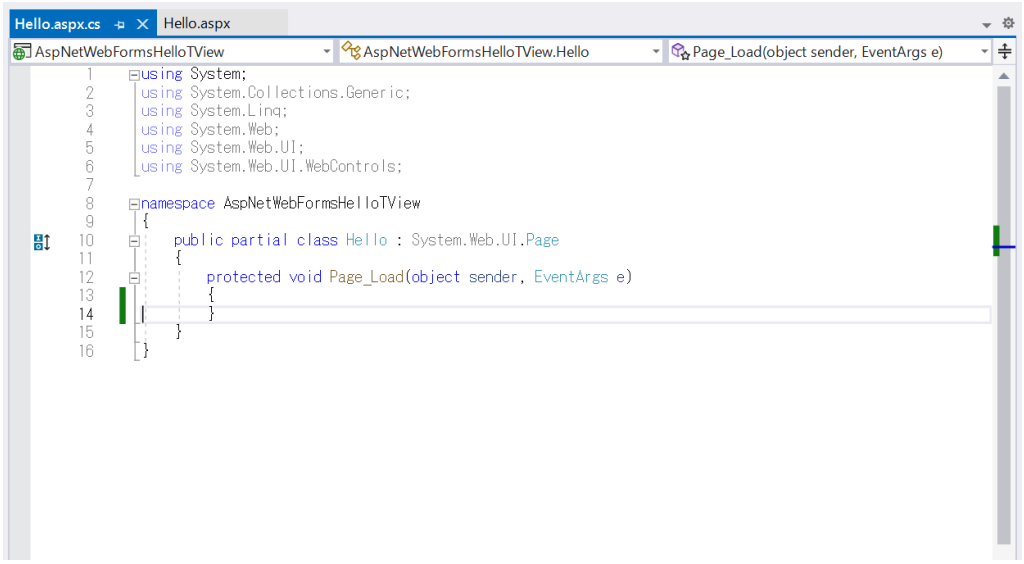


5. ブラウザが起動し、タイムビュー が組み込まれた Web ページが表示されます。



5.1.5 Page_Load イベント で タイムビューの初期表示設定を行う

1. Web フォームを「ソース」タブへ切り替えます。



列見出しを設定する

1. Page_Load イベントへ次のコードを追加します。

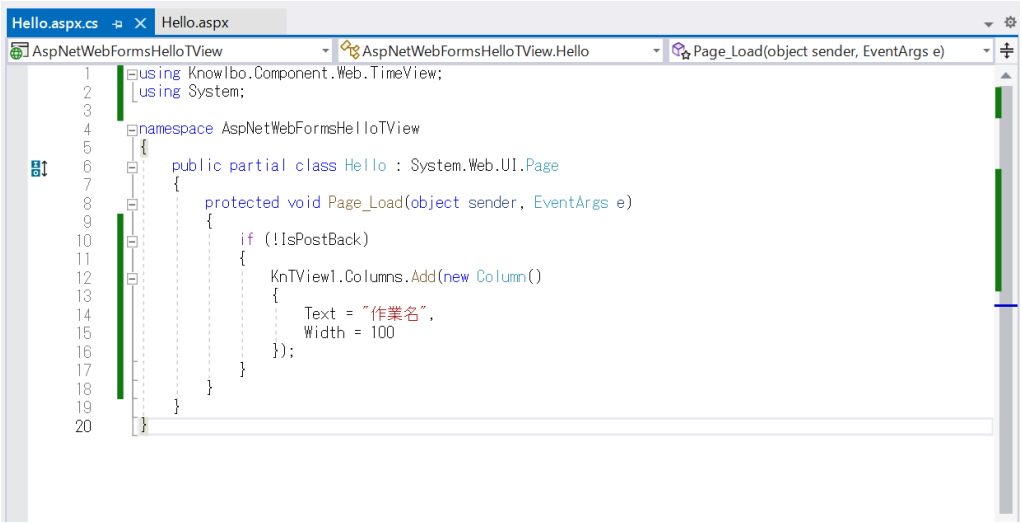
   このコードは タイムビュー に列見出しを追加し、列名を「作業名」、列幅を 100 ピクセルに設定しています。

```
if (!IsPostBack)
{
    KnTView1.Columns.Add(new Column()
    {
```

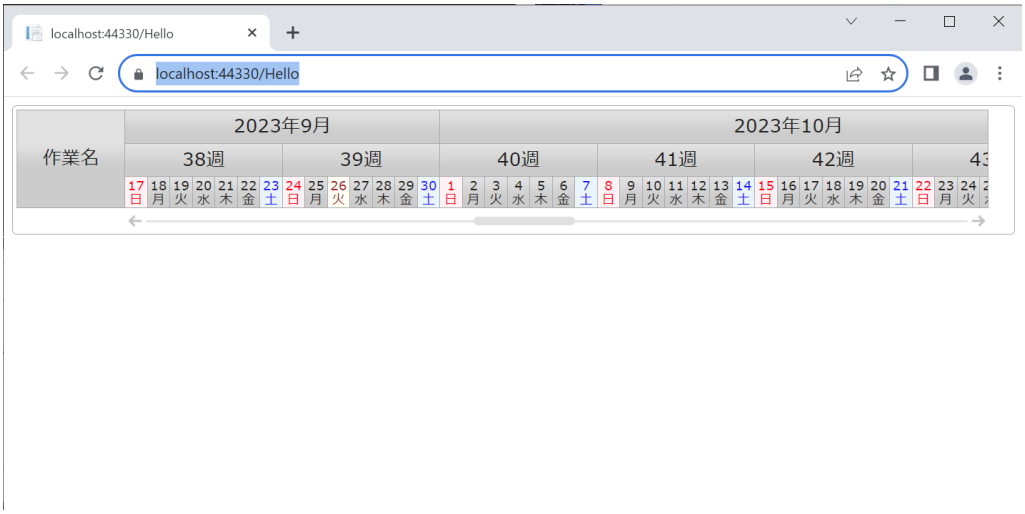
(次のページに続く)

(前のページからの続き)

```
        Text = "作業名",
        Width = 100
    });
}
```



2. F5 キーを押して列見出しが表示されることを確認します。

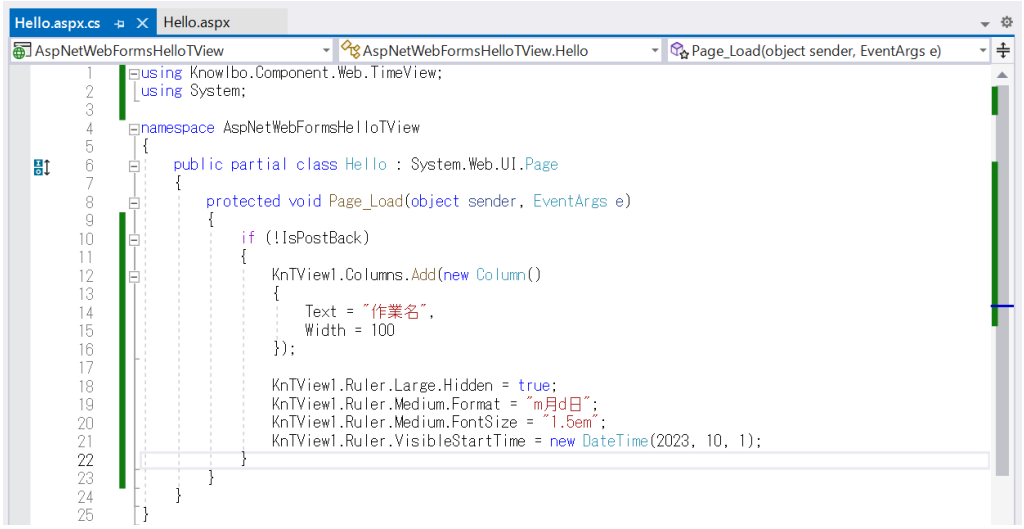


時間軸（タイム ルーラー）を設定する

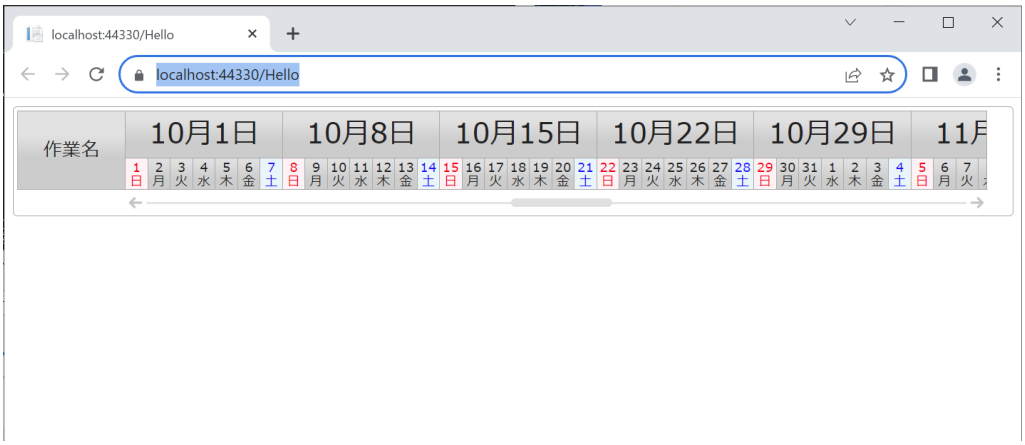
1. デバッグ実行を終了し、次は下記のコードを追加して時間軸の表示設定を行います。

   このコードは 大区分を非表示にし、既定で週ごとの表示を行う中区分を大きめの文字で「m 月 d 日」の表記とし、目盛りの左端の表示位置を 2023 年 10 月 1 日に設定します。

```
KnTView1.Ruler.Large.Hidden = true;
KnTView1.Ruler.Medium.Format = "m 月 d 日";
KnTView1.Ruler.Medium.FontSize = "1.5em";
KnTView1.Ruler.VisibleStartTime = new DateTime(2023, 10, 1);
```



2. F5 キーを押すと、時間軸（タイム ルーラーとも呼びます）が指定した表示に変わったことを確認できます。



アイテム と ピース で データ を表示する

1. デバッグ実行を終了し、次はデータを表示する設定を行います。

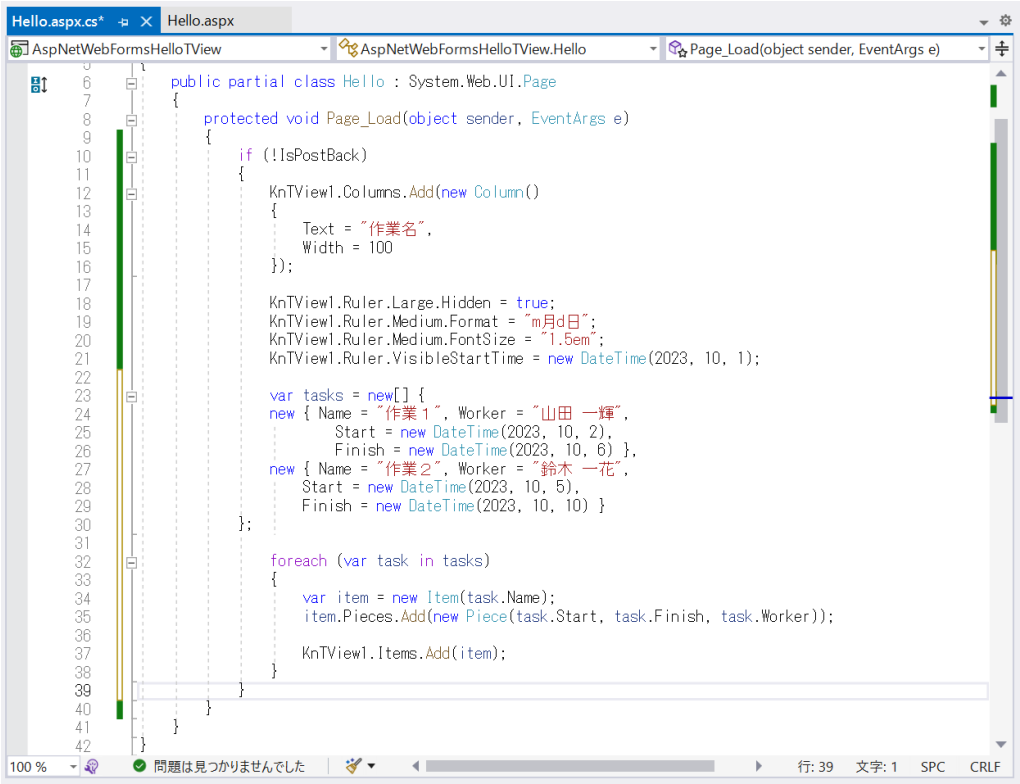
   このコードは tasks 配列の要素ごとにタイムビューの行を表す「アイテム」を作成し、「ピース」で作業期間と作業担当者を表示するように設定します。

```
var tasks = new[] {
    new { Name = "作業１", Worker = "山田 一輝",
            Start = new DateTime(2023, 10, 2),
            Finish = new DateTime(2023, 10, 6) },
    new { Name = "作業２", Worker = "鈴木 一花",
        Start = new DateTime(2023, 10, 5),
        Finish = new DateTime(2023, 10, 10) }
};

foreach (var task in tasks)
```

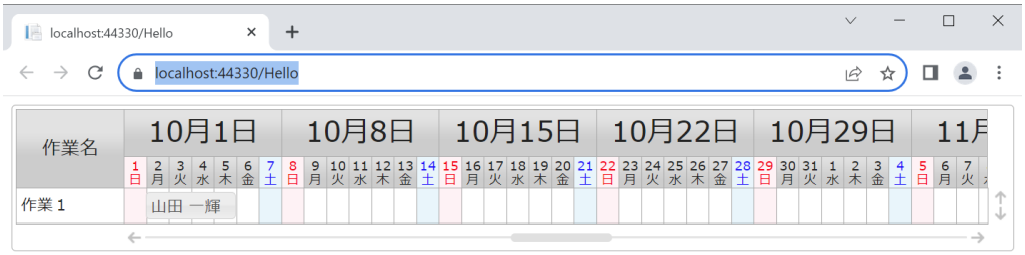
(次のページに続く)

(前のページからの続き)

```
{
    var item = new Item(task.Name);
    item.Pieces.Add(new Piece(task.Start, task.Finish, task.Worker));

    KnTView1.Items.Add(item);
}
```

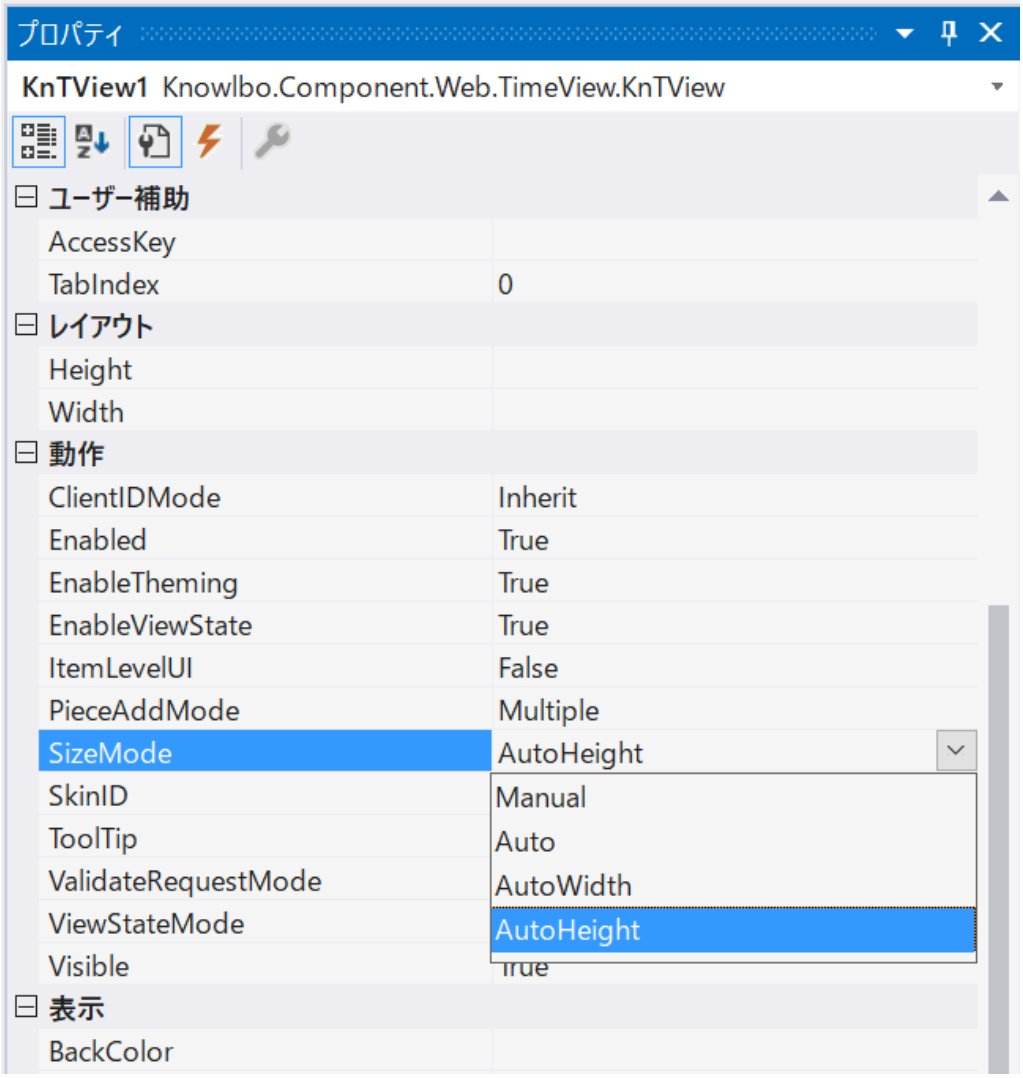


2. F5 キーを押して、tasks 配列の内容を タイムビュー を使って表示できることを確認します。

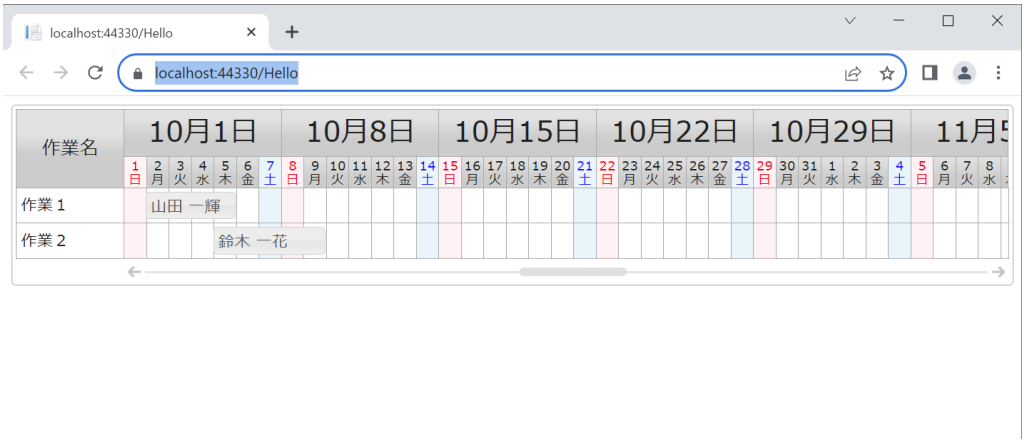


5.1.6 プロパティウィンドウ から タイムビュー のプロパティを設定する

1. デバッグ実行を終了し、「デザイナ」タブに切り替えます。
2. タイムビュー を選択して プロパティウィンドウを表示します。
3. 「SizeMode」プロパティを Manual から AutoHeight （自動ですべてのアイテムが見えるようにタイムビューの高さを調整する）へ変更します。

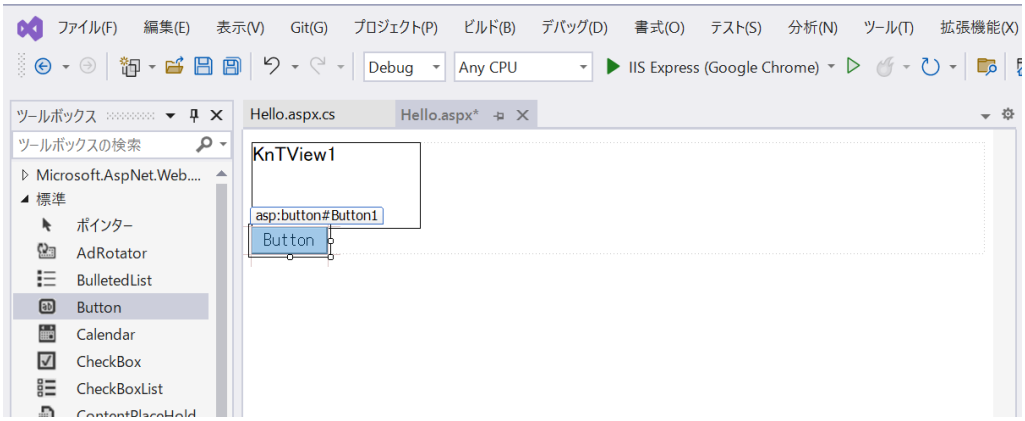


4. F5 キーを押して、先ほどまで一部表示が隠れていたアイテムがすべて見えるようになったことを確認します。



5.1.7 ボタン押下で動的に タイムビュー へデータを追加する

1. 「ツールボックス」で「Button」を選択して、Web フォーム上の任意の場所へドラッグアンドドロップします。



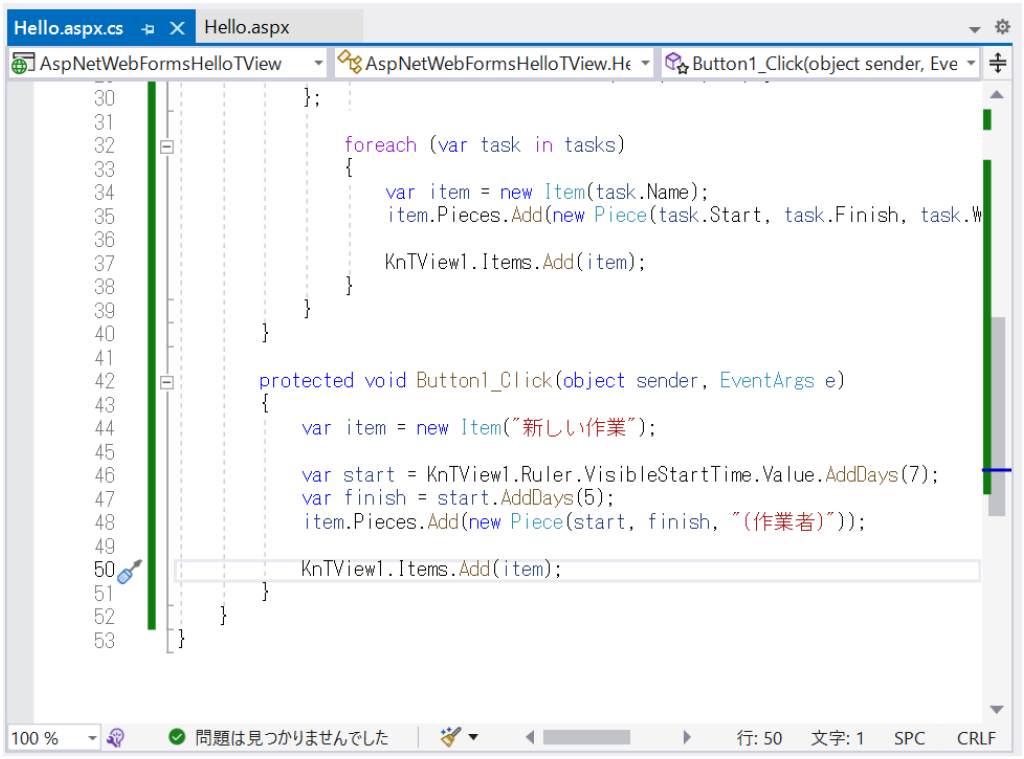
2. 追加したボタンのプロパティウィンドウにて、Text プロパティに「新しい作業」などの表示名を設定します。

3. 追加したボタンをダブルクリックして Click イベント ハンドラを追加します。

4. 下記のコードを追加します。

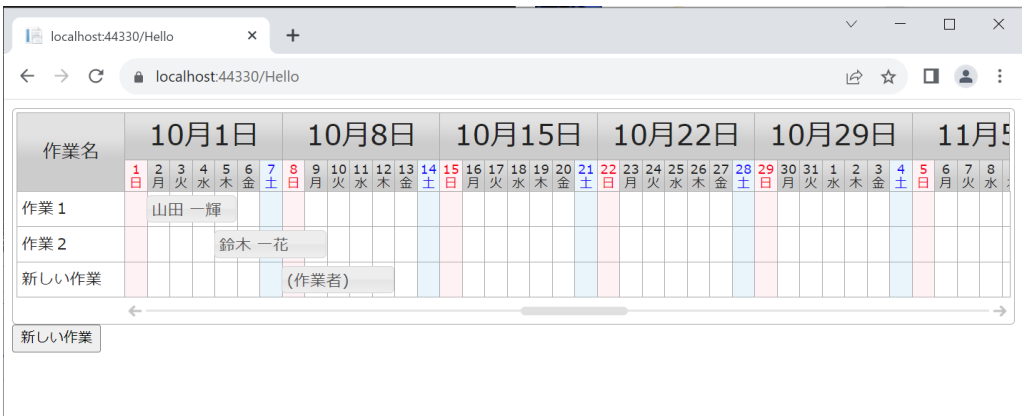
   このコードは タイムビューの末尾に新しいアイテム行を追加して表示名を「新しい作業」とし、現在の時間軸の左より 7 日後から 5 日分の長さを持つピースを追加します。

```
var item = new Item("新しい作業");

var start = KnTView1.Ruler.VisibleStartTime.Value.AddDays(7);
var finish = start.AddDays(5);
item.Pieces.Add(new Piece(start, finish, "(作業者)"));

KnTView1.Items.Add(item);
```



5. F5 キーを押して実行します。

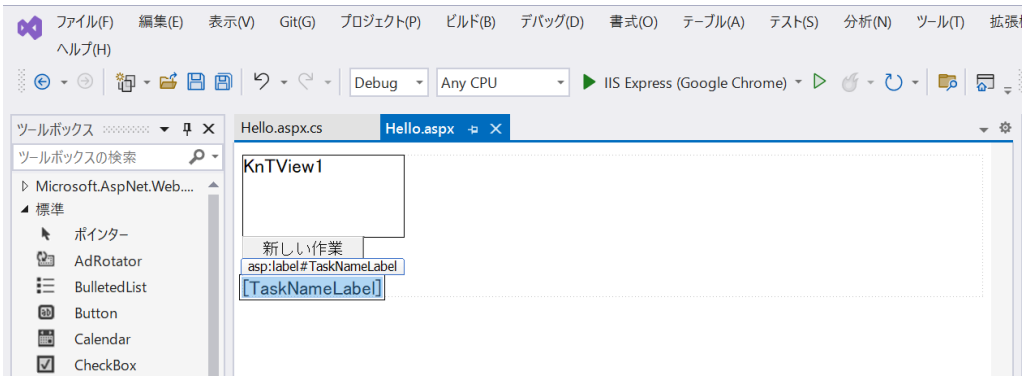
6. 「新しい作業」ボタンをクリックすると、アイテムが追加されることが確認できます。



5.1.8 タイムビュー が発生するイベントでデータを表示する

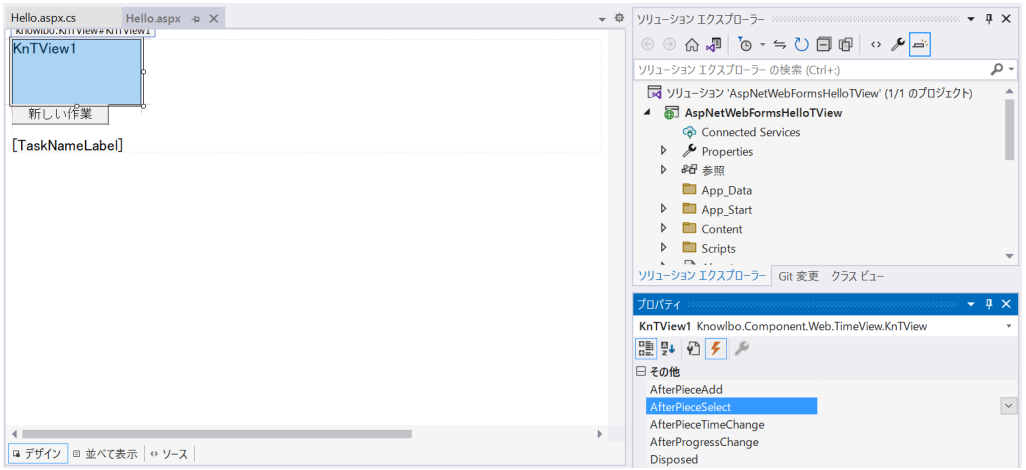
1. 「ツールボックス」で「Label」を選択して、Web フォーム上の任意の場所へドラッグアンドドロップします。

2. 追加したラベルのプロパティウィンドウにて、Text プロパティを空に、(ID) に「TaskNameLabel」を設定します。



3. 「KnTView1 (タイムビュー)」を選択して、プロパティウィンドウを☒（イベント一覧）へ切り替えます。

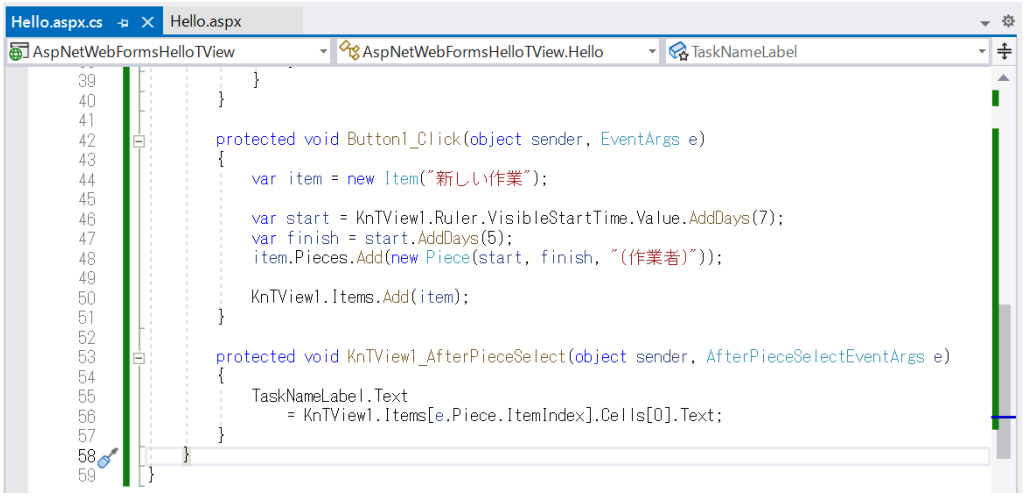
4. AfterPieceSelect イベントをダブルクリックします。



5. AfterPieceSelect イベント ハンドラへ下記のコードを追加します。

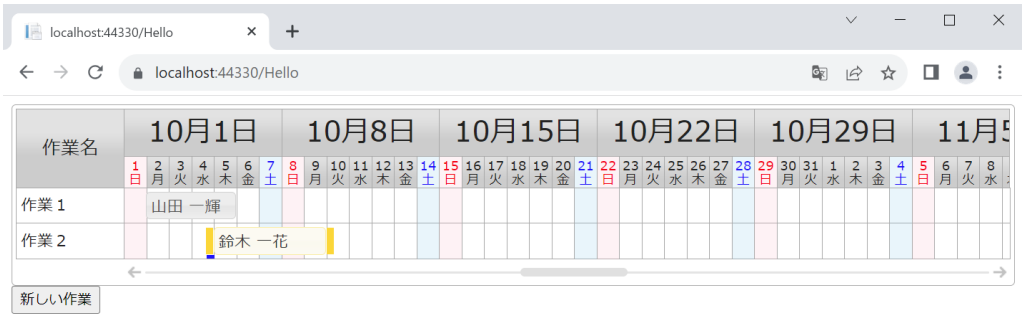
   このコードは クリックで選択したピースが位置するアイテムの 1 列目に格納している作業名をラベルに表示します。

```
TaskNameLabel.Text
    = KnTView1.Items[e.Piece.ItemIndex].Cells[0].Text;
```



6. F5 キーを押して実行します。

7. 任意のピースをクリックして選択すると、対応する作業名がラベルに表示されることを確認できます。



作業 2

5.1.9 JavaScript(jQuery) を用いてクライアントでも操作する場合

TimeView ASP.NET WebForms コントロールを通常のサーバーサイドでのプログラミングに加えて、クライアントの JavaScript (jQuery) でも操作したい場合は、PageRequestManager の pagedLoaded イベント内に記載するようにします。

下記の例では、ピース追加 UI への応答をサーバーへの AfterPieceAdd ポストバック イベントを用いずにクライアントの afterpieceadd イベントを設定してクライアント上で jQuery UI のモーダルダイアログによる業務上のそのほかの内容を入力するフォームを表示したあとにポストバックを発生させて、サーバー側にまとめて情報を送信してピース及びスケジュールデータの追加を行うようにしています。

約束事として、TimeView ASP.NET WebForms コントロールに対応するクライアント側の TimeView jQuery コントロールを参照するには、
$KnTVjQuery(タイムビューの HTML 要素の ID) で取得します。

```
<script type="text/javascript">
    const pageReqMgr = Sys.WebForms.PageRequestManager.getInstance();

    // 非同期通信完了後の処理
    pageReqMgr.add_pageLoaded(function (sender, args) {

        // タイムビュー WebForms コントロールが生成した
        // クライアント側のタイムビュー jQuery コントロールを取得
        const $timeview = $KnTVjQuery('#KnTView1');

        $timeview.timeview("option", {
            afterpieceadd: function (event, piece) {

                const item = items.item(piece.itemIndex());
```

(次のページに続く)

(前のページからの続き)

```
                // 開く
                $('#TaskInputItemIndex').val(piece.itemIndex());
                $('#TaskInputID').val("");
                $('#TaskInputName').val(nameCell.text());
                $('#TaskInputStart').val(formatForInputDate(piece.start()));
                $('#TaskInputFinish').val(formatForInputDate(addDays(piece.
↪finish(), -1)));
                $('#TaskInputWorker').val("");
                $('#TaskInputProgress').val("0");

                taskInputModalInstance.show();
            }
        });

        // 作業追加ボタンをクリック
        $("#TaskInputModal button.btn-modal-ok").on("click", function () {
            console.log("taskInputModal btn-modal-ok");

            if (!$('#MainForm')[0].checkValidity()) {
                return false;
            }
            if ($('#TaskInputStart').val() > $('#TaskInputFinish').val()) {
                $('#TaskInputFinish').addClass('is-invalid');
                $('#TaskInputFinish').next().text("終了日は開始日以降を設定して
ください。");
                return false;
            }

            taskInputModalInstance.hide();

            $('#TaskInputPostButton').trigger('click');
            return false;
        });
    });
```

5.2 TimeView jQuery コントロールを用いた開発

TimeView for Web の jQuery コントロールを使用した ASP.NET Core MVC アプリケーションの作成をはじめる手順を示します。

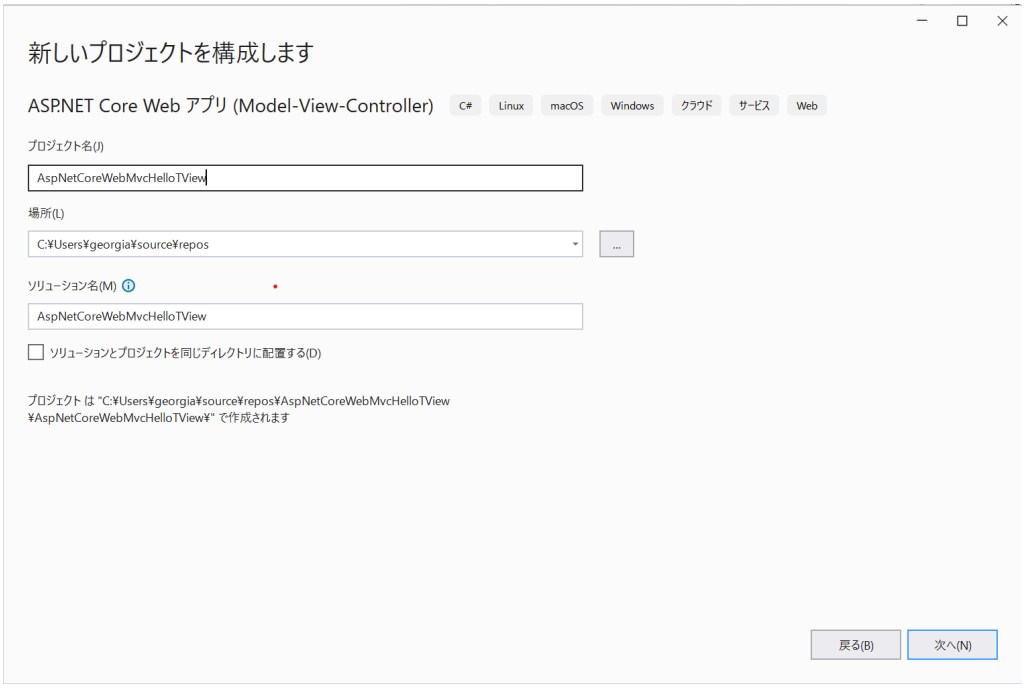
この例では Visual Studio Community 2022 を使用しています。

目次

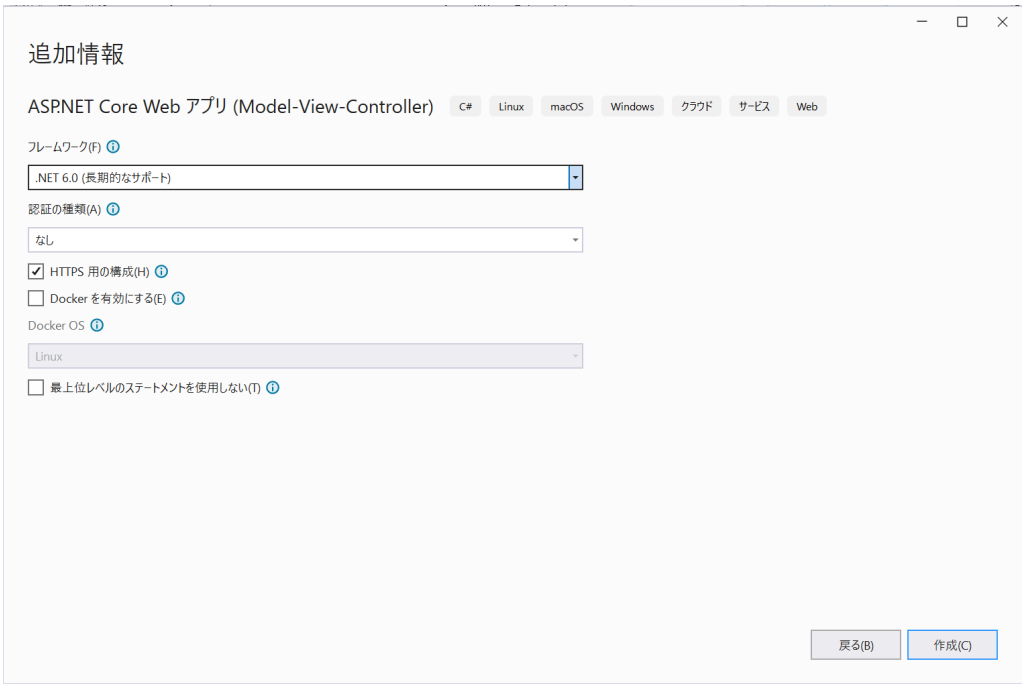
5.2.1 新しい ASP.NET Core MVC アプリケーション プロジェクトを作成する

1. 新しいプロジェクトとして「ASP.NET Core Web アプリ (Model-View-Controller) C# 」を選択します。

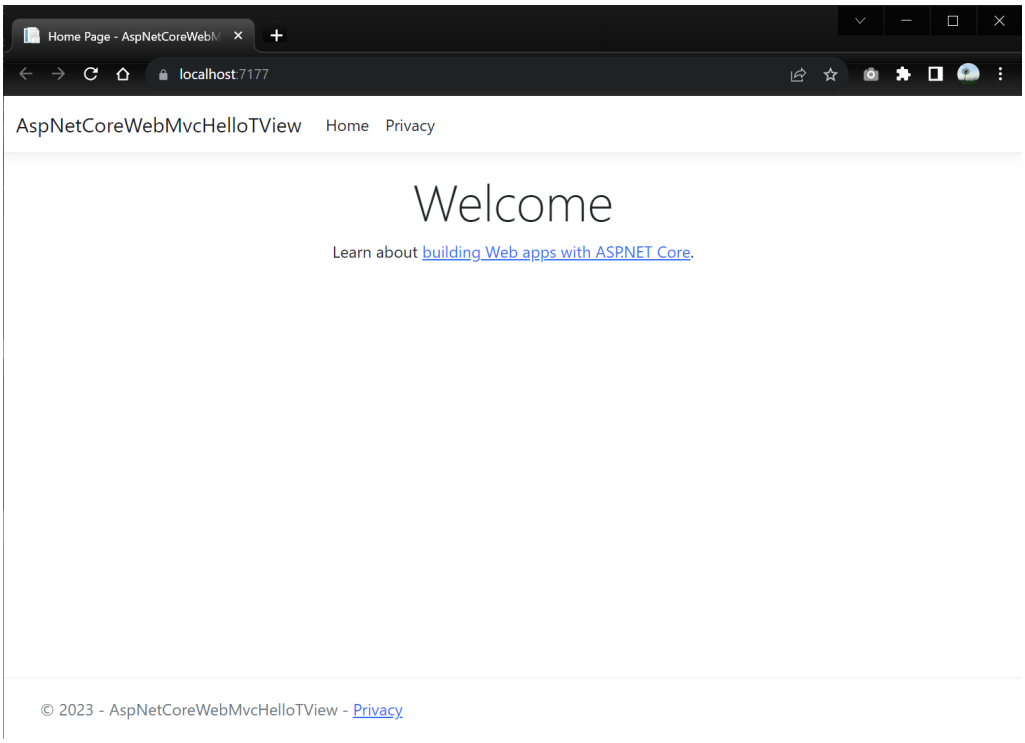
2. プロジェクトの構成情報を任意に設定して（既定の状態でも構いません）「次へ」ボタンをクリックします。



3. プロジェクトの追加情報を任意に設定して（既定の状態でも構いません）「作成」ボタンをクリックします。



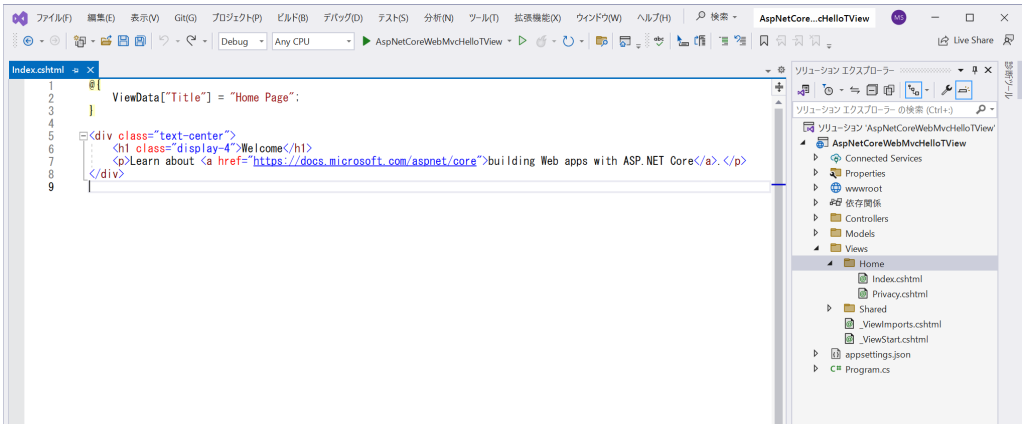
4. デバッグ実行 (F5 キー押下) して、タイムビュー追加前のアプリの状態を確認します。



5. 確認したらブラウザを閉じてデバッグを終了します。

5.2.2 Razor ページ に タイムビュー jQuery コントロール を追加する

1. Views/Home/Index.cshtml を開きます。

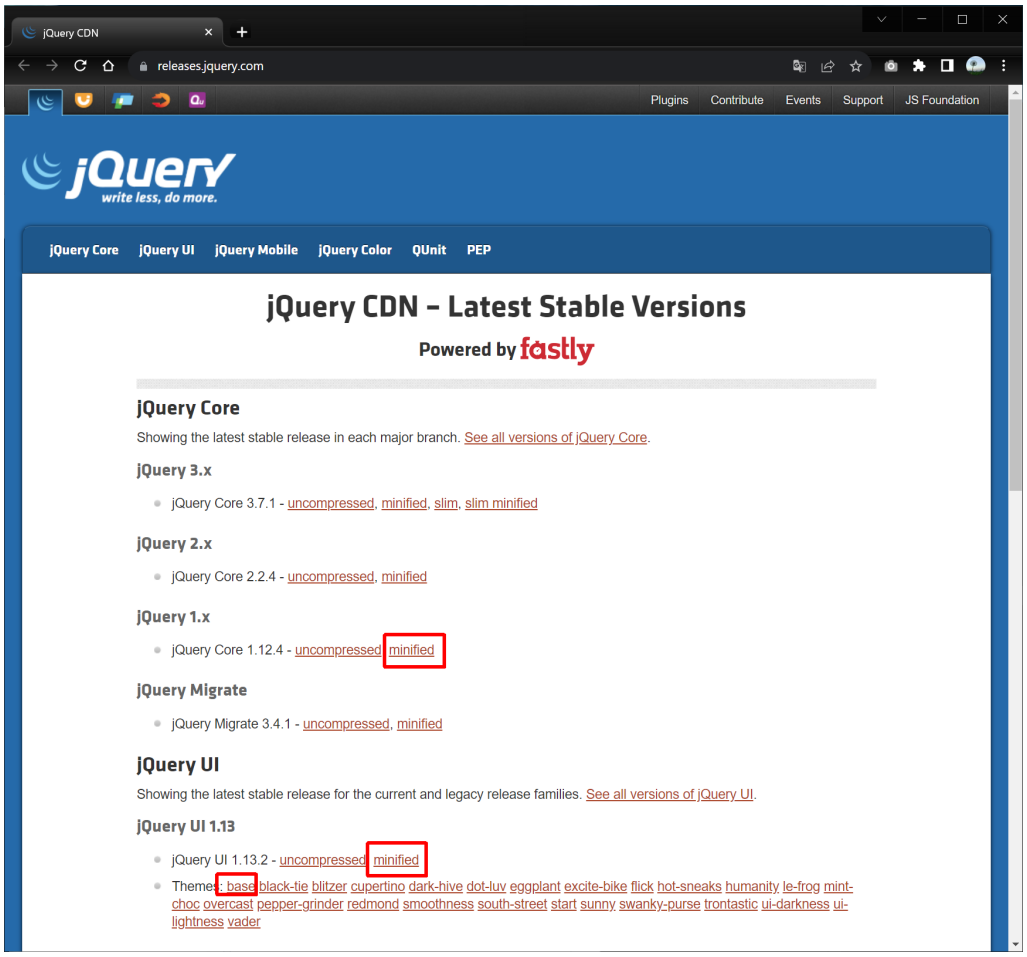


jQuery、および jQuery UI ライブラリの読み込みをページに追加する

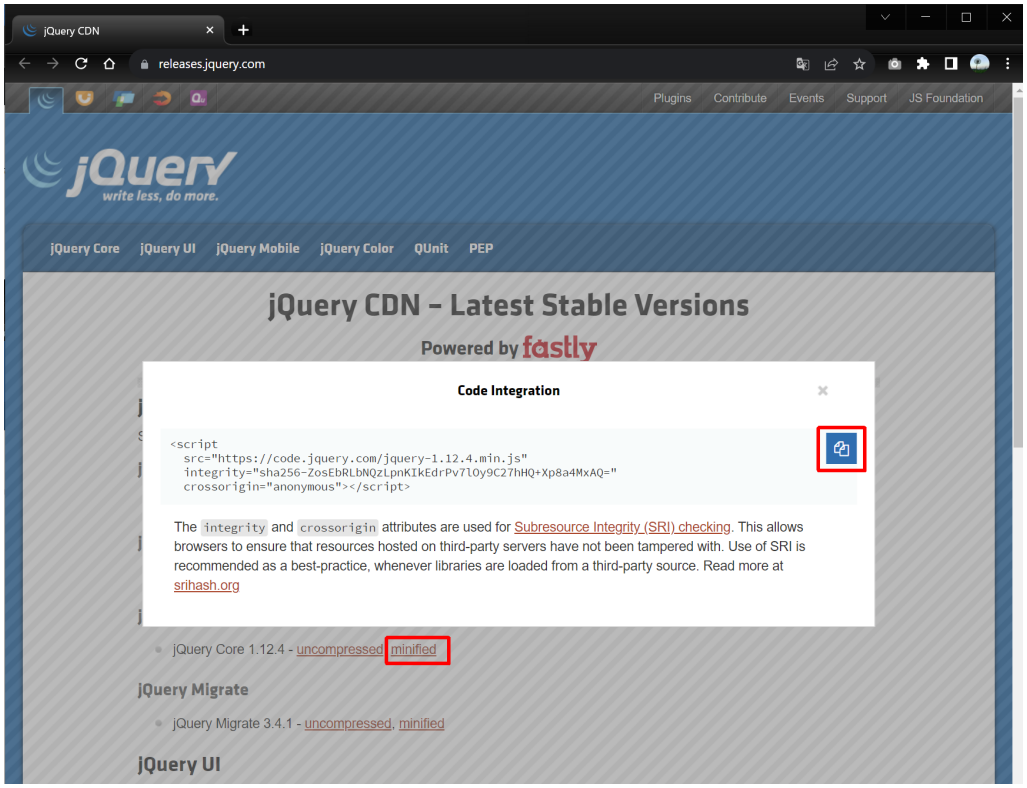
jQuery の公式サイトから タイムビュー jQuery コントロールが対応する jQuery、および jQuery UI ライブラリの JavaScript および CSS（スタイルシート）を読み込むタグを追加します。

1. jQuery の公式 CDN サイト を開きます。

   https://releases.jquery.com/

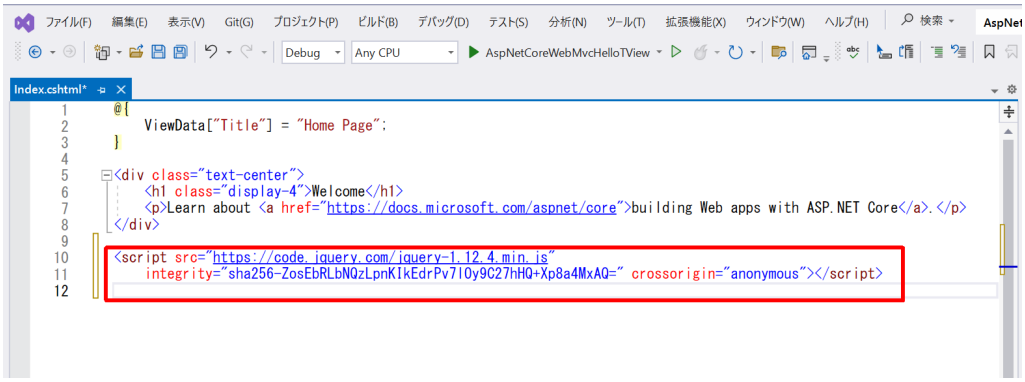


2. jQuery 1.12.4 minified をクリックして script 読み込みタグをコピーします。

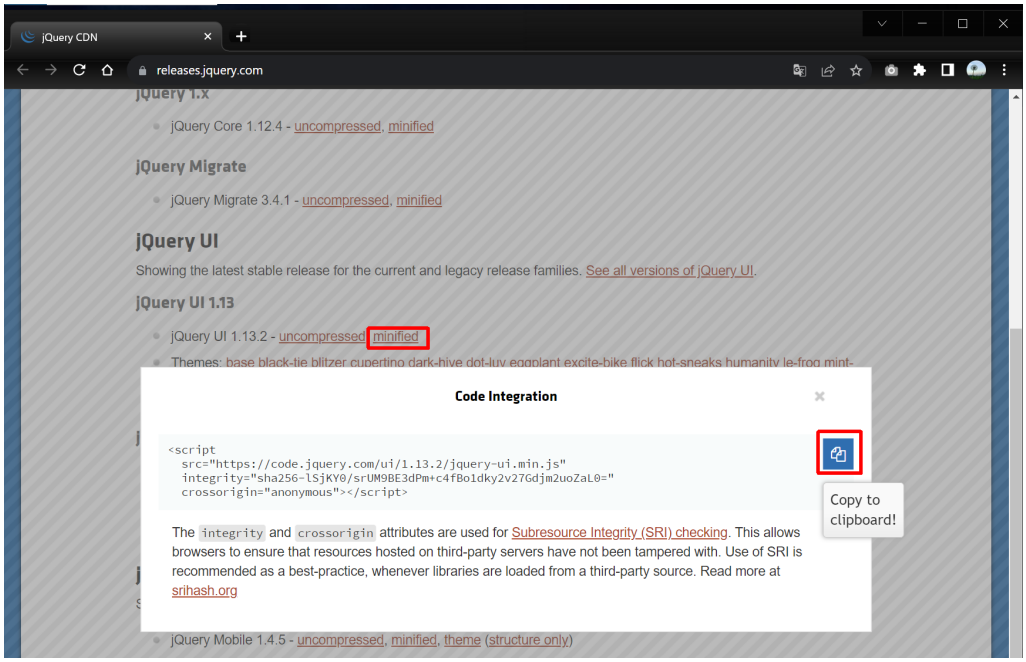


```
<script src="https://code.jquery.com/jquery-1.12.4.min.js"
    integrity="sha256-ZosEbRLbNQzLpnKIkEdrPv7lOy9C27hHQ+Xp8a4MxAQ="
    crossorigin="anonymous"></script>
```

3. index.cshtml の末尾へ貼り付けます。



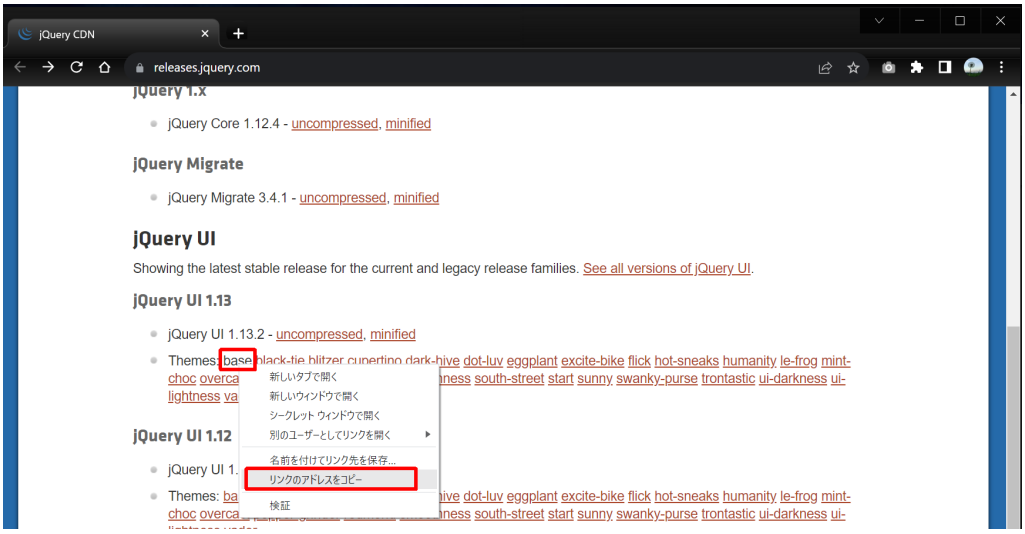
4. jQuery UI 1.13.2 minified をクリックして script 読み込みタグをコピーします。




```
<script src="https://code.jquery.com/ui/1.13.2/jquery-ui.min.js"
    integrity="sha256-lSjKY0/srUM9BE3dPm+c4fBo1dky2v27Gdjm2uoZaL0="
    crossorigin="anonymous"></script>
```

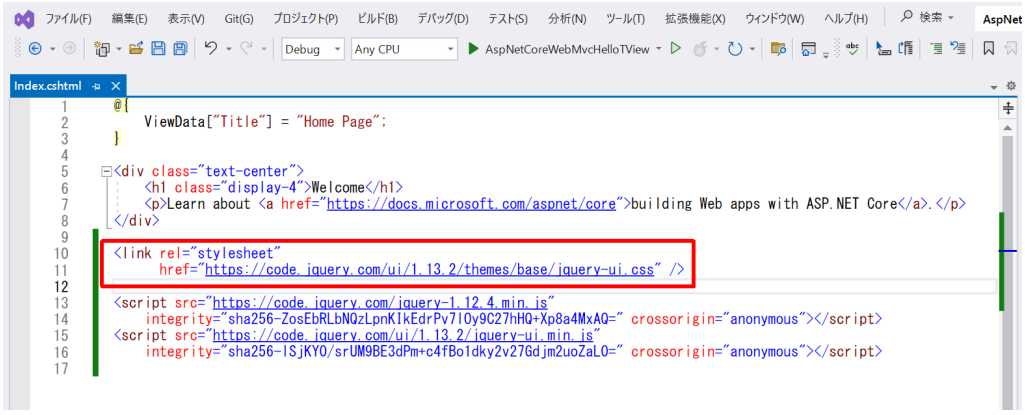
5. index.cshtml の末尾へ貼り付けます。

6. jQuery UI 1.13 の 任意のテーマ CSS の リンクを右クリックして URL をコピーします。




```
https://code.jquery.com/ui/1.13.2/themes/base/jquery-ui.css
```

7. index.cshtml の <script>タグ より前に、link タグ を追加して href 属性に 上記 URL を貼り付けます。

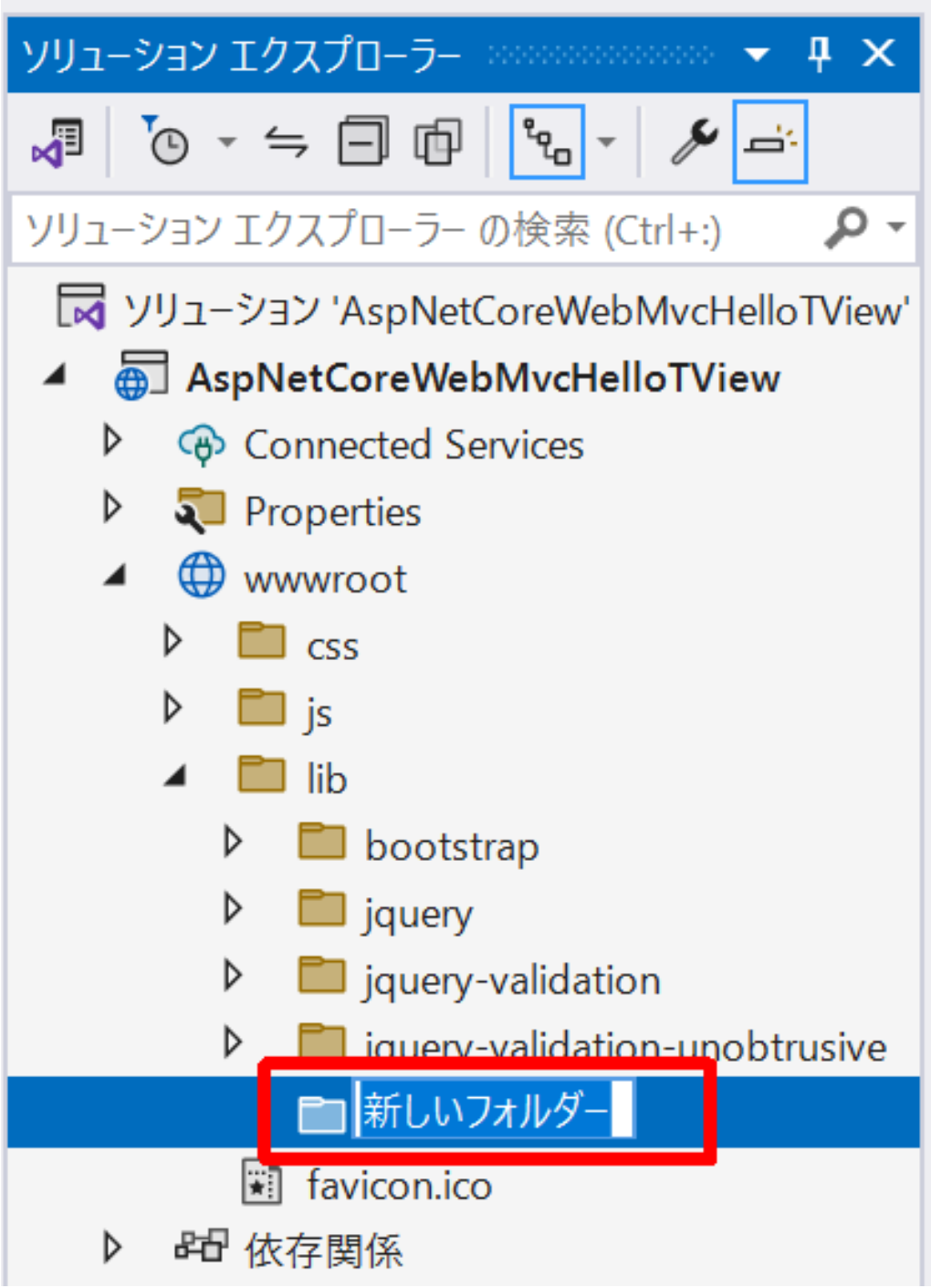



```
<link rel="stylesheet"
    href="https://code.jquery.com/ui/1.13.2/themes/base/jquery-ui.css" /
↪>
```

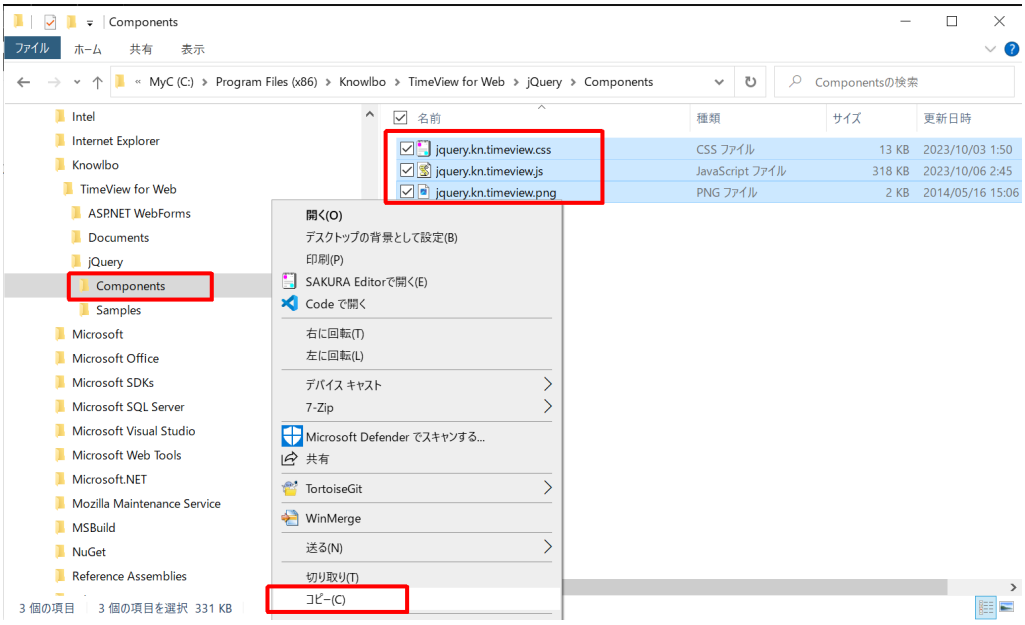
タイムビュー jQuery ライブラリの読み込みをページに追加する

タイムビュー for Web インストールフォルダ から タイムビュー jQuery コントロールのファイル一式をプロジェクトへコピーして、読み込むタグをページへ追加します。

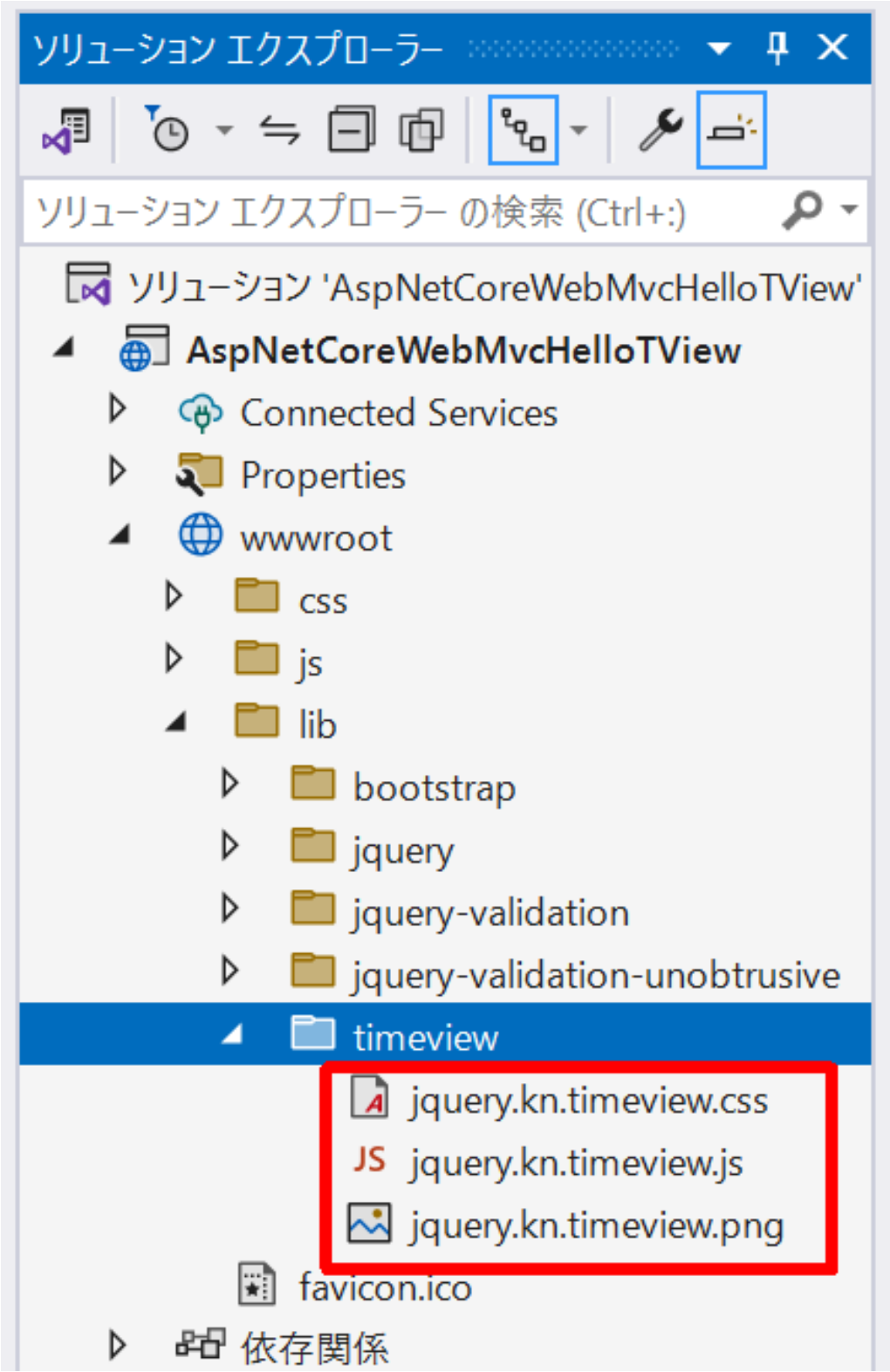
1. プロジェクトの wwwroot/lib フォルダへ 新しいサブフォルダー「timeview」を作成します（この作業は必須ではありません。分かりやすくする便宜上の任意作業です）



2. タイムビュー for Web インストールフォルダ の jQuery/Components フォルダのファイルすべてをコピーします。



3. コピーしたファイルをプロジェクトの上記で作成したフォルダ (wwwroot/lib/timeview) を右クリックして貼り付けます。



4. index.cshtml の末尾へ タイムビュー jQuery コントロール の JavaScript ライブラリを読み込む下記のタグを追加します。

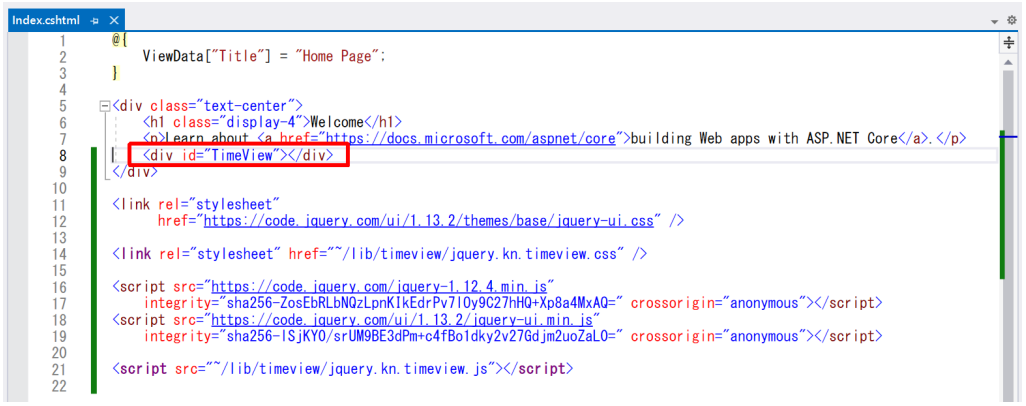

```
<script src="~/lib/timeview/jquery.kn.timeview.js"></script>
```

5. index.cshtml の jQuery UI のスタイルシートを読み込部タグの下に タイムビュー jQuery コントロール の スタイルシートを読み込む下記のタグを追加します。


```
<link rel="stylesheet" href="~/lib/timeview/jquery.kn.timeview.css" />
```

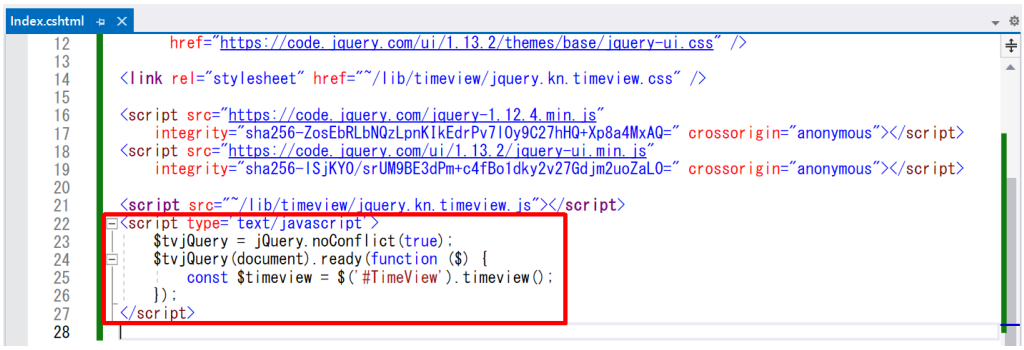
タイムビュー jQuery コントロールをページに追加する

1. タイムビュー jQuery コントロールの作成先となる div タグを追加します（div タグの id 属性値は「timeview」である必要はなく任意の命名でもちろん構いません）



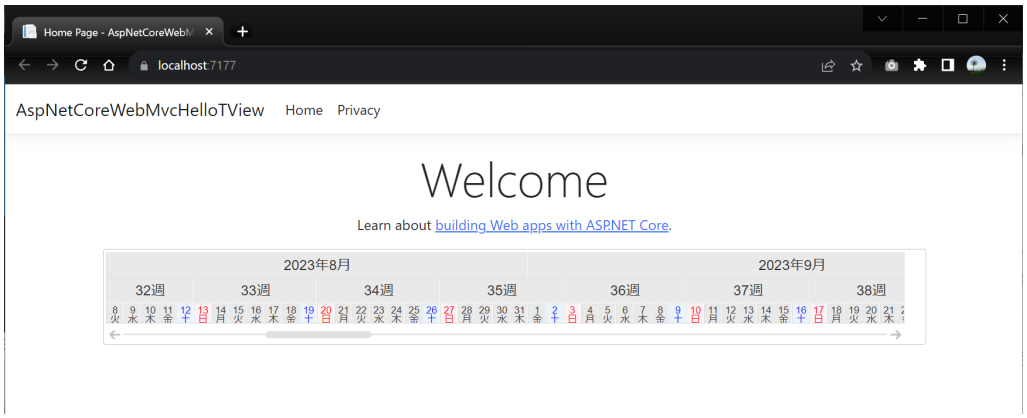
```
<div id="TimeView"></div>
```

2. タイムビュー jQuery コントロールを作成する下記の JavaScript コードを timeview.js ライブラリ読み込みタグの直下に追加します。




```
<script type='text/javascript'>
    $tvjQuery = jQuery.noConflict(true);
    $tvjQuery(document).ready(function ($) {
        const $timeview = $('#TimeView').timeview();
    });
</script>
```

3. デバッグ実行 (F5 キー押下) して、追加したタイムビューが表示されることを確認します。



4. 確認したらブラウザを閉じてデバッグを終了します。

5. ここまで編集した index.cshtml ページのコード全体は下記のとおりになります。

```
@{
    ViewData["Title"] = "Home Page";
}

<div class="text-center">
    <h1 class="display-4">Welcome</h1>
    <p>Learn about <a href="https://docs.microsoft.com/aspnet/core">
↪building Web apps with ASP.NET Core</a>.</p>
    <div id="TimeView"></div>
```

(次のページに続く)

(前のページからの続き)

```
</div>

<link rel="stylesheet"
    href="https://code.jquery.com/ui/1.13.2/themes/base/jquery-ui.css" /
↪>

<link rel="stylesheet" href="~/lib/timeview/jquery.kn.timeview.css" />

<script src="https://code.jquery.com/jquery-1.12.4.min.js"
    integrity="sha256-ZosEbRLbNQzLpnKIkEdrPv7lOy9C27hHQ+Xp8a4MxAQ="
    crossorigin="anonymous"></script>
<script src="https://code.jquery.com/ui/1.13.2/jquery-ui.min.js"
    integrity="sha256-lSjKY0/srUM9BE3dPm+c4fBo1dky2v27Gdjm2uoZaL0="
    crossorigin="anonymous"></script>

<script src="~/lib/timeview/jquery.kn.timeview.js"></script>
<script type='text/javascript'>
    $tvjQuery = jQuery.noConflict(true);
    $tvjQuery(document).ready(function ($) {
        const $timeview = $('#TimeView').timeview();
    });
</script>
```

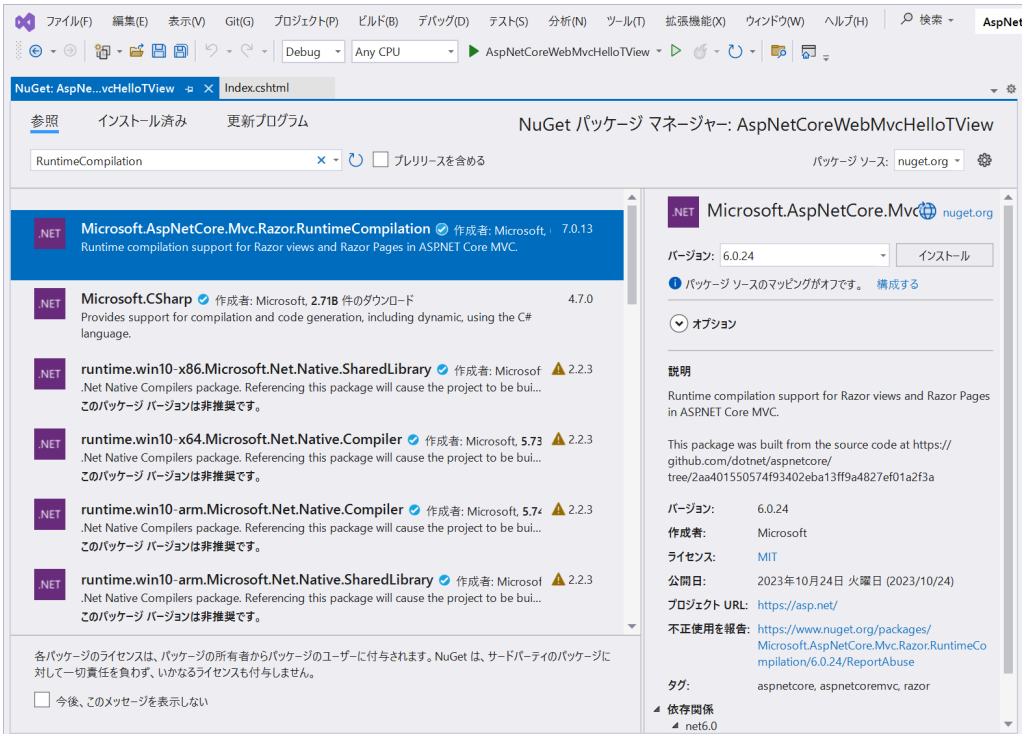
5.2.3 アプリの実行中に Razor ページの更新を反映できるようにする

タイムビュー jQuery コントロールの使用方法とは直接関係ありませんが、この節で示す設定を行うことにより、デバッグ実行したまま Razor ページの更新をブラウザのリロードのみで確認できるようになります。

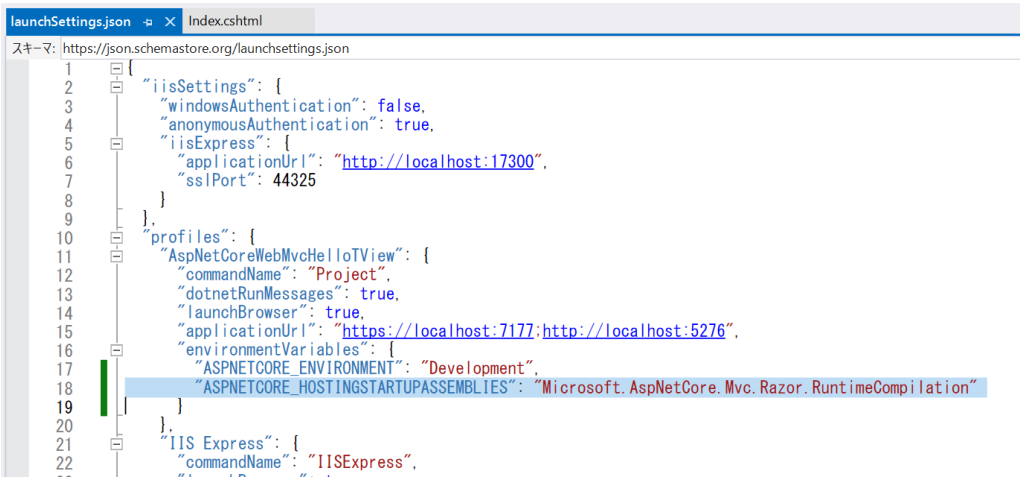
参考文献：ASP.NET Core での Razor ファイルのコンパイル | Microsoft Learn

1. Microsoft.AspNetCore.Mvc.Razor.RuntimeCompilation NuGet パッケージをインストールします。

> **注釈:** インストールするバージョンは、使用する .NET のメジャーバージョンに合わせるようにします。例えばプロジェクトが .NET 6.0 を使用している場合は 当 NuGet パッケージも 6.0.x をインストールするようにします。



2. launchSettings.json で起動プロファイルの environmentVariables セクションを変更します。




```
{
    "profiles": {
        "AspNetCoreWebMvcHelloTView": {
        "commandName": "Project",
        "dotnetRunMessages": true,
        "launchBrowser": true,
        "applicationUrl": "https://localhost:7177;http://localhost:5276
↪",
        "environmentVariables": {
            "ASPNETCORE_ENVIRONMENT": "Development",
            "ASPNETCORE_HOSTINGSTARTUPASSEMBLIES": "Microsoft.
↪AspNetCore.Mvc.Razor.RuntimeCompilation"
```

(次のページに続く)

(前のページからの続き)

```
        }
    },
}
```

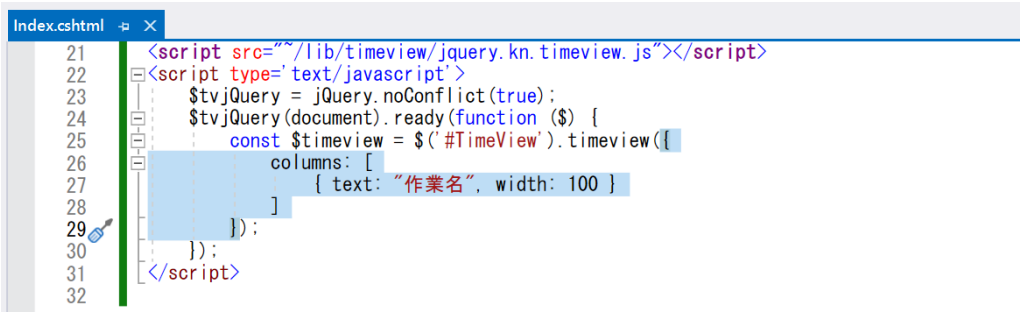
5.2.4 タイムビューの初期表示設定を行う

列見出しを設定する

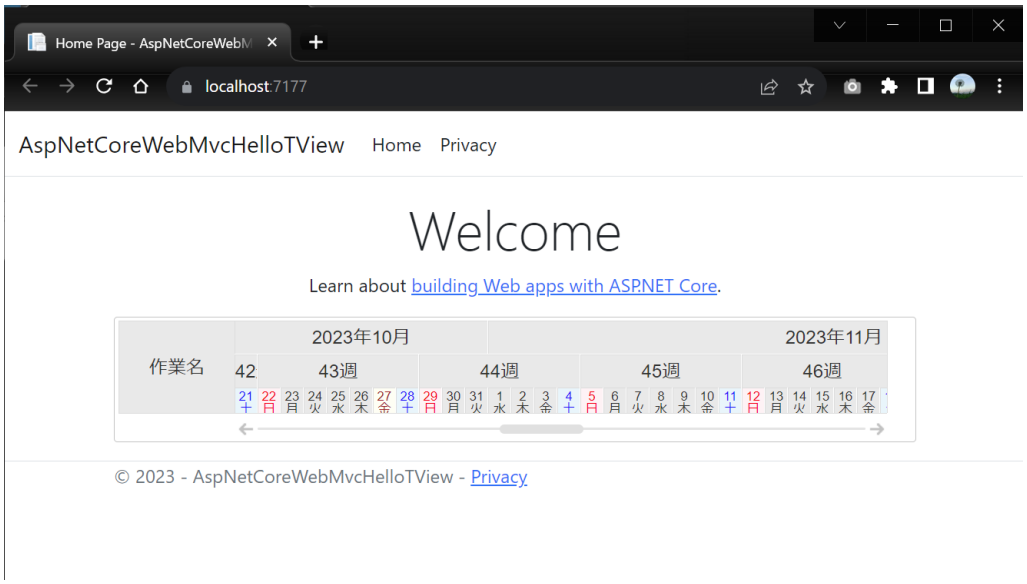
1. timeview 関数の引数に下記の json データを追加します。

   このコードは タイムビュー に列見出しを追加し、列名を「作業名」、列幅を 100 ピクセルに設定しています。

```
    $tvjQuery(document).ready(function ($) {
        const $timeview = $('#TimeView').timeview({
            columns: [
                { text: "作業名", width: 100 }
            ]
        });
    });
```



2. ページを更新すると列見出しが表示されることを確認できます。



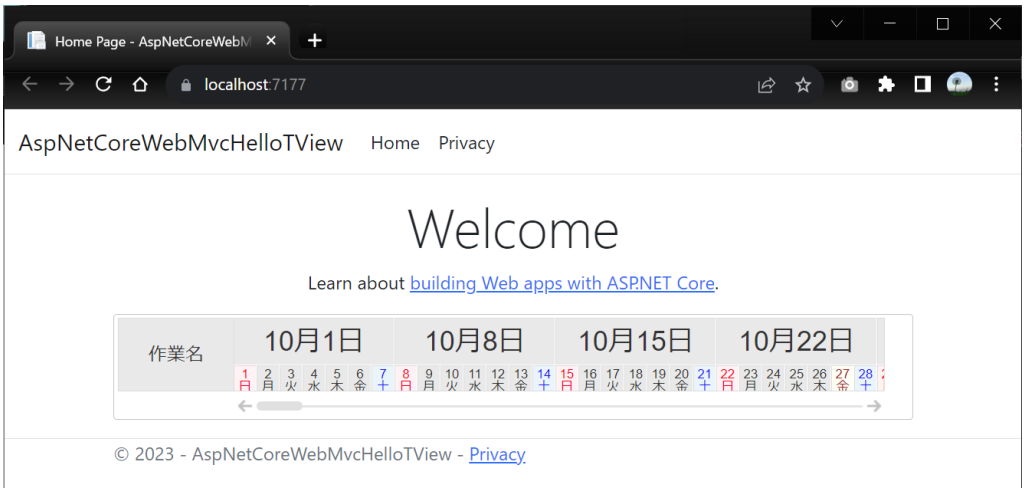
時間軸（タイム ルーラー）を設定する

1. 次は下記のコードを追加して時間軸の表示設定を行います。

   このコードは 大区分を非表示にし、既定で週ごとの表示を行う中区分を大きめの文字で「m 月 d 日」の表記とし、目盛りの表示期間を 2023 年 10 月 1 日から 1 年間に設定します。

```
$tvjQuery(document).ready(function ($) {
    const $timeview = $('#TimeView').timeview({
        timeRuler: {
            large: {
                hidden: true
            },
            medium: {
                fontSize: "1.5em",
                unit: "week",
                format: "m 月 d 日"
            },
            start: '2023/10/1',
            finish: '2024/10/1',
        },
        columns: [
            { text: "作業名", width: 100 }
        ]
    });
});
```

2. ページを更新すると、時間軸（タイム ルーラーとも呼びます）が指定した表示に変わったことを確認できます。



アイテム と ピース で データ を表示する

1. 次はデータを表示する設定を行います。

   このコードは tasks 配列の要素ごとにタイムビューの行を表す「アイテム」を作成し、「ピース」で作業期間と作業担当者を表示するように設定します。また、「sizeMode」プロパティを autoHeight （すべてのアイテムが見えるようにタイムビューの高さを自動調整する）へ変更します。

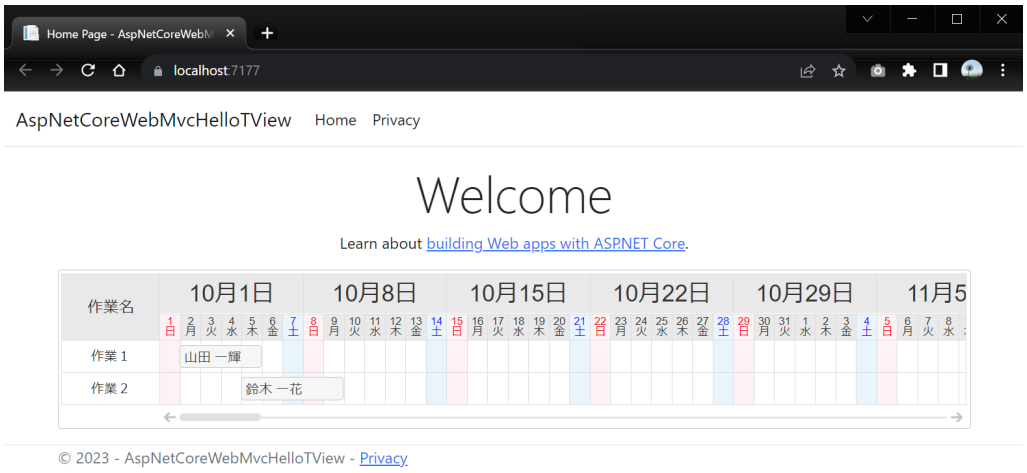
```
$tvjQuery(document).ready(function ($) {
    const $timeview = $('#TimeView').timeview({
        timeRuler: {
            large: {
                hidden: true
            },
            medium: {
                fontSize: "1.5em",
                unit: "week",
                format: "m 月 d 日"
            },
            start: '2023/10/1',
            finish: '2024/10/1',
        },
        columns: [
            { text: "作業名", width: 100 }
        ],
        sizeMode: "autoHeight"
    });

    const tasks = [
        {
            name: "作業１", worker: "山田 一輝",
```

(次のページに続く)

(前のページからの続き)

```
                start: "2023/10/2", finish: "2023/10/6"
            },
            {
                name: "作業２", worker: "鈴木 一花",
                start: "2023/10/5", finish: "2023/10/10"
            }
        ];

        tasks.forEach((task) => {

            const tvItem = $timeview.timeview("addItem");
            tvItem.cells().item(0).text(task.name);

            const tvPiece = $timeview.timeview("addPiece", {
                itemIndex: tvItem.index(),
                start: task.start, finish: task.finish,
                caption: task.worker
            });
        });
    });
```

2. F5 キーを押して、tasks 配列の内容を タイムビュー を使って表示できることを確認します。



5.2.5 ボタン押下で動的に タイムビュー へデータを追加する

1. タイムビューを表示する div タグの下に input タグでボタンを追加します。

```
<div id="TimeView"></div>
<div>
    <input id="NewTaskButton" type="button" value="新しい作業" />
</div>
```

2. 追加したボタンの Click イベント ハンドラを追加します。

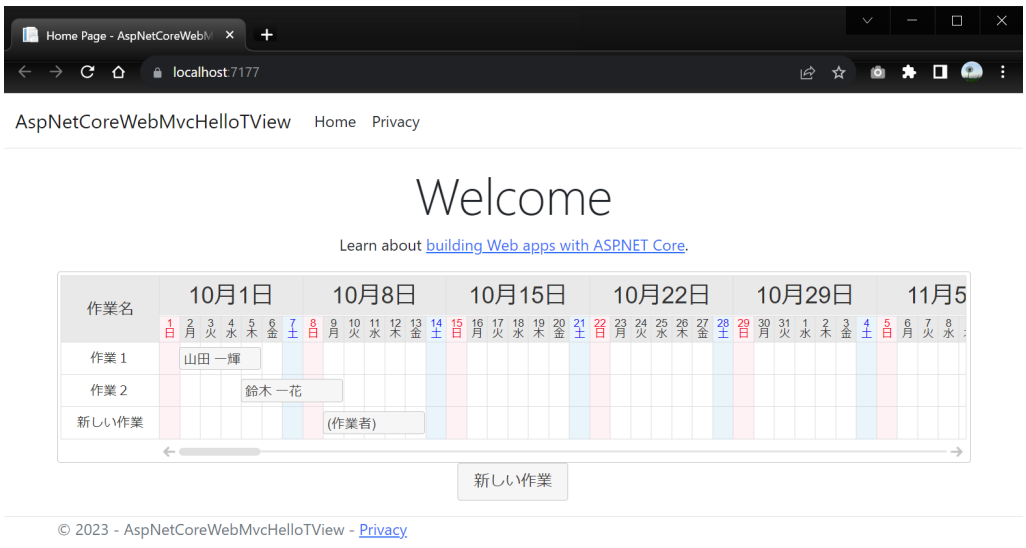
   このコードは タイムビューの末尾に新しいアイテム行を追加し、表示名を「新しい作業」とし、現在の時間軸の左より 8 日後から 5 日分の長さを持つピースを追加します。

```
$("#NewTaskButton").button().click(function () {

    const tvRuler = $timeview.timeview("ruler");
    const visibleStartTime = tvRuler.visibleStartTime();

    var taskStart = new Date(
        visibleStartTime.getFullYear(), visibleStartTime.getMonth(),
        visibleStartTime.getDate() + 8);
    var taskFinish = new Date(
        taskStart.getFullYear(), taskStart.getMonth(),
        taskStart.getDate() + 5);

    const tvItem = $timeview.timeview("addItem", {
        text: "新しい作業"
    });

    const tvPiece = $timeview.timeview("addPiece", {
        itemIndex: tvItem.index(),
        start: taskStart,
        finish: taskFinish,
        caption: "(作業者)"
    });
});
```

```
Index.cshtml
58
59          tasks.forEach((task) => {
60
61              const tvItem = $timeview.timeview("addItem");
62              tvItem.cells().item(0).text(task.name);
63
64              const tvPiece = $timeview.timeview("addPiece", {
65                  itemIndex: tvItem.index(),
66                  start: task.start, finish: task.finish,
67                  caption: task.worker
68              });
69          });
70
71          $("#NewTaskButton").button().click(function () {
72
73              const tvRuler = $timeview.timeview("ruler");
74              const visibleStartTime = tvRuler.visibleStartTime();
75
76              var taskStart = new Date(
77                  visibleStartTime.getFullYear(), visibleStartTime.getMonth(),
78                  visibleStartTime.getDate() + 8);
79              var taskFinish = new Date(
80                  taskStart.getFullYear(), taskStart.getMonth(),
81                  taskStart.getDate() + 5);
82
83              const tvItem = $timeview.timeview("addItem", {
84                  text: "新しい作業"
85              });
86
87              const tvPiece = $timeview.timeview("addPiece", {
88                  itemIndex: tvItem.index(),
89                  start: taskStart,
90                  finish: taskFinish,
91                  caption: "(作業者)"
92              });
93          });
94      });
95  </script>
96
```

3. F5 キーを押して Web ページをリロードします。

4. 「新しい作業」ボタンをクリックすると、アイテムが追加されることが確認できます。



5.2.6 タイムビュー が発生するイベントでデータを表示する

1. 「新しい作業」ボタンの下に、作業名を表示するため下記のタグを追加します。

```
<div>
    <p>作業名：<span id="TaskNameLabel"></span></p>
</div>
```

Index.cshtml

```
@{
    ViewData["Title"] = "Home Page";
}

<div class="text-center">
    <h1 class="display-4">Welcome</h1>
    <p>Learn about <a href="https://docs.microsoft.com/aspnet/core">building Web apps with ASP.NE
    <div id="TimeView"></div>
    <div>
        <input id="NewTaskButton" type="button" value="新しい作業" />
    </div>
    <div>
        <p>作業名：<span id="TaskNameLabel"></span></p>
    </div>
</div>
```

2. timeview 関数を用いて、option の afterpieceselect イベント ハンドラを設定します。

   このコードは クリックで選択したピースが位置するアイテムの 1 列目に格納している作業名をラベルに表示します。

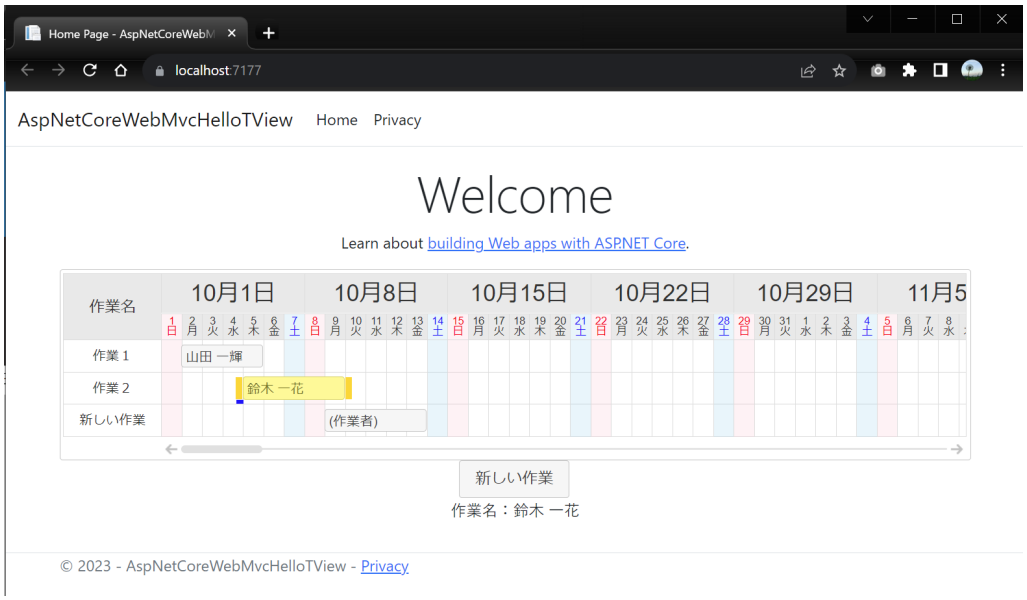
```
$timeview.timeview("option", {
    afterpieceselect: function (event, piece) {
        $("#TaskNameLabel").text(piece.caption().text());
    }
});
```

Index.cshtml

```
                visibleStartTime.getFullYear(), visibleStartTime.getMonth(),
                visibleStartTime.getDate() + 8);
            var taskFinish = new Date(
                taskStart.getFullYear(), taskStart.getMonth(),
                taskStart.getDate() + 5);

            const tvItem = $timeview.timeview("addItem", {
                text: "新しい作業"
            });

            const tvPiece = $timeview.timeview("addPiece", {
                itemIndex: tvItem.index(),
                start: taskStart,
                finish: taskFinish,
                caption: "(作業者)"
            });
        });

        $timeview.timeview("option", {
            afterpieceselect: function (event, piece) {
                $("#TaskNameLabel").text(piece.caption().text());
            }
        });
    });
</script>
```

3. F5 キーを押してページの更新を読み込みます。

4. 任意のピースをクリックにて選択すると、対応する作業名がラベルに表示されることを確認できます。



5. ここまで編集した index.cshtml ページのコード全体は下記のとおりになります。

```
@{
    ViewData["Title"] = "Home Page";
}

<div class="text-center">
    <h1 class="display-4">Welcome</h1>
    <p>Learn about <a href="https://docs.microsoft.com/aspnet/core">
↪building Web apps with ASP.NET Core</a>.</p>
    <div id="TimeView"></div>
    <div>
        <input id="NewTaskButton" type="button" value="新しい作業" />
    </div>
    <div>
        <p>作業名：<span id="TaskNameLabel"></span></p>
    </div>
</div>

<link rel="stylesheet"
    href="https://code.jquery.com/ui/1.13.2/themes/base/jquery-ui.css" /
↪>

<link rel="stylesheet" href="~/lib/timeview/jquery.kn.timeview.css" />

<script src="https://code.jquery.com/jquery-1.12.4.min.js"
        integrity="sha256-ZosEbRLbNQzLpnKIkEdrPv7lOy9C27hHQ+Xp8a4MxAQ="␣
↪crossorigin="anonymous"></script>
<script src="https://code.jquery.com/ui/1.13.2/jquery-ui.min.js"
        integrity="sha256-lSjKY0/srUM9BE3dPm+c4fBo1dky2v27Gdjm2uoZaL0="␣
```

(次のページに続く)

(前のページからの続き)

```
↪crossorigin="anonymous"></script>

<script src="~/lib/timeview/jquery.kn.timeview.js"></script>
<script type='text/javascript'>
    $tvjQuery = jQuery.noConflict(true);
    $tvjQuery(document).ready(function ($) {
        const $timeview = $('#TimeView').timeview({
            timeRuler: {
                large: {
                    hidden: true
                },
                medium: {
                    fontSize: "1.5em",
                    unit: "week",
                    format: "m 月 d 日"
                },
                start: "2023/10/1",
                finish: "2024/10/1",
            },
            columns: [
                { text: "作業名", width: 100 }
            ],
            sizeMode: "autoHeight"
        });

        const tasks = [
            {
                name: "作業１", worker: "山田 一輝",
                start: "2023/10/2", finish: "2023/10/6"
            },
            {
                name: "作業２", worker: "鈴木 一花",
                start: "2023/10/5", finish: "2023/10/10"
            }
        ];

        tasks.forEach((task) => {

            const tvItem = $timeview.timeview("addItem");
            tvItem.cells().item(0).text(task.name);

            const tvPiece = $timeview.timeview("addPiece", {
                itemIndex: tvItem.index(),
                start: task.start, finish: task.finish,
                caption: task.worker
```

(次のページに続く)

(前のページからの続き)

```
            });
        });

        $("#NewTaskButton").button().click(function () {

            const tvRuler = $timeview.timeview("ruler");
            const visibleStartTime = tvRuler.visibleStartTime();

            var taskStart = new Date(
                visibleStartTime.getFullYear(), visibleStartTime.
↪getMonth(),
                visibleStartTime.getDate() + 8);
            var taskFinish = new Date(
                taskStart.getFullYear(), taskStart.getMonth(),
                taskStart.getDate() + 5);

            const tvItem = $timeview.timeview("addItem", {
                text: "新しい作業"
            });

            const tvPiece = $timeview.timeview("addPiece", {
                itemIndex: tvItem.index(),
                start: taskStart,
                finish: taskFinish,
                caption: "(作業者)"
            });
        });

        $timeview.timeview("option", {
            afterpieceselect: function (event, piece) {
                $("#TaskNameLabel").text(piece.caption().text());
            }
        });
    });
</script>
```

第6章　リファレンス

6.1 KnTView コントロール

タイム ビュー コントロールは、さまざまな時間スケジュール情報をガント チャート形式で表示します。

6.1.1 概要

タイム ビュー コントロールは、時間を基準に計画された情報の入力や効果的な表示に用いることができます。タイム ビュー コントロールでは、仕事や予定などをピースという単位で表します。ピースの入力や移動は、マウスを使って直接画面上で行うことが可能です。横軸にタイム スケールという時間の目盛りを持ち、目盛りの単位は分単位から年単位まで指定が可能です。

ファイル名

Knowlbo.Component.Web.TimeView.dll

名前空間

Knowlbo.Componen.Web.TimeView

クラス名

KnTView

6.1.2 例

次のコード例は、Web ページでタイムビュー コントロールを作成する方法を示します。

```
<%@ Register Assembly="Knowlbo.Component.Web.TimeView" Namespace="Knowlbo.Component.
↪Web.TimeView" TagPrefix="knowlbo" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title></title>
    <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
    <script src="https://code.jquery.com/ui/1.13.2/jquery-ui.min.js"></script>
    <link href="https://code.jquery.com/ui/1.13.2/themes/redmond/jquery-ui.css" rel=
↪"stylesheet" type="text/css" />
</head>
<body>
```

(次のページに続く)

(前のページからの続き)

```
    <form id="form1" runat="server">
        <knowlbo:KnTView ID="KnTView1" runat="server" />
    </form>
</body>
</html>
```

6.1.3 プロパティ一覧

タイムビュー コントロール独自

プロパティ	内容
Columns	列見出しのコレクションを取得します。
ItemLevelUI	アイテムの展開/折りたたみをマウスで行えるようにするかを取得または設定します。
Items	行アイテムのコレクションを取得します。
PieceAddMode	各アイテムにマウス操作によってピースをいくつ追加できるようにするかを取得または設定します。
Ruler	タイムルーラーを取得します。
Selection	選択データ集を取得します。
SizeMode	コントロールの表示サイズの決め方を取得または設定します。
SpecialTimes	特別時間帯コレクションを示します。
SumPane	集計表示ペインを取得します。

WebControl から継承

プロパティ	内容
AccessKey	(TODO: 要動作確認) タイムビュー コントロールにすばやく移動できるアクセス キーを取得または設定します (継承元 WebControl)
Attributes	タイムビュー コントロールに対応するプロパティがない HTML 属性をまとめて取得または設定します (継承元 WebControl)
BackColor	タイムビュー コントロールの背景色を取得または設定します (継承元 WebControl)
BorderColor	タイムビュー コントロールの境界線の色を取得または設定します (継承元 WebControl)
BorderStyle	タイムビュー コントロールの境界線スタイルを取得または設定します (継承元 WebControl)
BorderWidth	タイムビュー コントロールの境界線の幅を取得または設定します (継承元 WebControl)
ControlStyle	タイムビュー コントロールのスタイルを取得します (継承元 WebControl)
ControlStyleCreated	タイムビュー コントロールの Style オブジェクトが ControlStyle プロパティに対して作成されたかどうかを示す値を取得します (継承元 WebControl)
CssClass	タイムビュー コントロールに関連付けられた CSS クラスの名前を取得または設定します (継承元 WebControl)
Enabled	(TODO: 要動作実装) タイムビュー コントロールを有効にするかどうかを示す値を取得または設定します (継承元 WebControl)
EnableTheming	テーマがこのコントロールに適用されるかどうかを示す値を取得または設定します (継承元 WebControl)
Font	タイムビュー コントロールに関連付けられたフォント プロパティを取得します (継承元 WebControl)
ForeColor	タイムビュー コントロールの前景色を取得または設定します (継承元 WebControl)
HasAttributes	タイムビュー コントロールに属性セットがあるかどうかを示す値を取得します (継承元 WebControl)
Height	タイムビュー コントロールの高さを取得または設定します (継承元 WebControl)
SkinID	タイムビュー コントロールに適用するスキンを取得または設定します (継承元 WebControl)
Style	Web サーバー コントロールの外側のタグにスタイル属性として表示されるテキスト属性のコレクションを取得します (継承元 WebControl)
SupportsDisabledAttribute	IsEnabled プロパティの値が false の場合にコントロールを無効な状態でレンダリングするかどうかを取得します (継承元 WebControl)
TabIndex	Web サーバー コントロールのタブ インデックスを取得または設定します (継承元 WebControl)
TagKey	コントロール タグの名前を取得します。 このプロパティは、主にコントロールの開発者によって使用されます (継承元 WebControl)
ToolTip	マウス ポインターが Web サーバー コントロールの上を移動したときに表示されるテキストを取得または設定します (継承元 WebControl)
Width	タイムビュー コントロールの幅を取得または設定します (継承元 WebControl)

Control から継承

プロパティ	内容
ClientID	タイムビュー コントロールに対して ASP.NET によって生成される HTML マークアップのコントロール ID を取得します (継承元 Control)
ClientIDMode	タイムビュー コントロールの ClientID プロパティの値を生成するために使用されるアルゴリズムを取得または設定します (継承元 Control)
EnableViewState	タイムビュー コントロールがそのビュー状態を保持するかどうかを取得または設定します (継承元 Control)
ID	タイムビュー コントロールに割り当てられたプログラム ID を取得または設定します (継承元 Control)
LoadViewStateByID	タイムビュー コントロールがインデックスではなく ID によりビューステートの読み込みを行うかどうかを示す値を取得します (継承元 Control)
NamingContainer	同じ ID プロパティ値を持つ複数のサーバー コントロールを区別するための一意の名前空間を作成する、サーバー コントロールの名前付けコンテナーへの参照を取得します (継承元 Control)
Page	サーバー コントロールを含んでいる Page インスタンスへの参照を取得します (継承元 Control)
Parent	ページ コントロールの階層構造における、サーバー コントロールの親コントロールへの参照を取得します (継承元 Control)
RenderingCompatibility	レンダリングされる HTML と互換性がある ASP.NET のバージョンを表す値を取得します (継承元 Control)
Site	デザイン サーフェイスに現在のコントロールを表示するときに、このコントロールをホストするコンテナーに関する情報を取得します (継承元 Control)
TemplateControl	このコントロールを格納しているテンプレートへの参照を取得または設定します (継承元 Control)
TemplateSourceDirectory	現在のサーバー コントロールを格納している Page または UserControl の仮想ディレクトリを取得します (継承元 Control)
UniqueID	階層構造で修飾されたサーバー コントロールの一意の ID を取得します (継承元 Control)
ValidateRequestMode	ブラウザーからのクライアント入力の安全性をコントロールで調べるかどうかを示す値を取得または設定します (継承元 Control)
ViewState	同一のページに対する複数の要求にわたって、サーバー コントロールのビューステートを保存し、復元できるようにする状態情報のディクショナリを取得します (継承元 Control)
ViewStateIgnoresCase	StateBag オブジェクトが大文字小文字を区別しないかどうかを示す値を取得します (継承元 Control)
ViewStateMode	このコントロールのビューステート モードを取得または設定します (継承元 Control)
Visible	ASP.NET がこのコントロールをレンダリングする必要があるかどうかを取得または設定します (継承元 Control)

6.1.4 メソッド一覧

タイムビュー コントロール独自

メソッド	内容
GetPieceByKey	すべてのピースの中から指定のピース キーと一致するピースを取得します。
GetPieceByValue	すべてのピースの中から指定のピース値と一致するピースを取得します。

WebControl から継承

メソッド	内容
ApplyStyle	指定したスタイルの空白以外の要素を Web コントロールにコピーして、コントロールの既存のスタイル要素を上書きします (継承元 WebControl)
CopyBaseAttributes	指定した Web サーバー コントロールから、Style オブジェクトでカプセル化されていないプロパティをこのメソッドの呼び出し元の Web サーバー コントロールにコピーします (継承元 WebControl)
MergeStyle	指定したスタイルの空白以外の要素を Web コントロールにコピーしますが、コントロールの既存のスタイル要素は上書きしません (継承元 WebControl)

Control から継承

メソッド	内容
ApplyStyleSheetSkin	ページのスタイル シートに定義されたスタイル プロパティをコントロールに適用します (継承元 Control)
Focus	(TODO: 要動作確認) コントロールに入力フォーカスを設定します (継承元 Control)

Object から継承

メソッド	内容
GetHashCode	既定のハッシュ関数として機能します (継承元 Object)
GetType	現在のインスタンスの Type を取得します (継承元 Object)

6.1.5 イベント一覧

タイムビュー コントロール独自

イベント	内容
AfterPieceAdd	ユーザー操作によってピースが追加されたことを通知します。
AfterPieceSelect	ピースが選択されたことを通知します。
AfterPieceTimeChange	ユーザー操作によってピースの開始日時または終了日時が変更されたことを通知します。
AfterProgressChange	ユーザー操作によってピースの達成率が変更されたことを通知します。
PieceDblClick	ピースがダブルクリックされたことを通知します。

Control から継承

イベント	内容
DataBinding	サーバー コントロールの DataBind メソッドが呼び出されることでデータソースにバインドされるときに発生します (継承元 Control)
Disposed	サーバー コントロールがメモリから解放されると発生します。これは、ASP.NET ページが要求されている場合のサーバー コントロールの有効期間における最終段階です (継承元 Control)
Init	サーバー コントロールが初期化されると発生します。これは、サーバー コントロールの有効期間における最初の手順です (継承元 Control)
Load	Init イベントの後にサーバー コントロールが Page オブジェクトに読み込まれると発生します (継承元 Control)
PreRender	サーバー コントロールのコンテンツがレンダリングを開始する直前に発生します (継承元 Control)
Unload	サーバー コントロールがメモリからアンロードされると発生します (継承元 Control)

6.1.6 プロパティ

Columns プロパティ

タイムビュー コントロールの列見出しのコレクションを取得します。

定義

public *Columns* Columns { get; }

プロパティ値

Columns クラス への参照です。

概要

Columns クラスへの参照を返します。実行時のみ使用できます。

ItemLevelUI プロパティ

タイムビュー コントロールのアイテムの展開/折りたたみをマウスで行えるようにするかを取得または設定します。

定義

public bool ItemLevelUI { get; set; }

プロパティ値

アイテムの展開/折りたたみをマウスで行えるようにする場合は true。それ以外の場合は false。 既定値は、false です。

Items プロパティ

タイムビュー コントロールの行アイテムのコレクションを取得します。

定義

public *Items* Items { get; }

プロパティ値

Items クラス への参照です。

概要

タイムビュー コントロールのアイテム (即ち行) の集合を表す、Items コレクションへの参照を返します。実行時のみ使用できます。アイテムの追加や削除は、Items プロパティから参照する *Items* コレクションのメソッドを使って行うことができます。

PieceAddMode プロパティ

各アイテムに対し、マウス操作によってピースをいくつ追加できるかを取得または設定します。

定義

public *PieceAddMode* 列挙型 PieceAddMode { get; set; }

プロパティ値

PieceAddMode 列挙型 値のいずれか 1 つ。 既定値は Multiple です。

Ruler プロパティ

タイムビュー コントロールのタイムルーラーを取得します。

定義

public *Ruler* Ruler { get; }

プロパティ値

TimeRuler クラス への参照です。

概要

タイム ビュー コントロールの右上部に表示する時間軸の定規を表す TimeRuler オブジェクトへの参照を返します。Ruler プロパティを使って、タイムルーラーに関するさまざまな設定を行うことができます。

Selection プロパティ

タイムビュー コントロールの選択データ集を取得します。

定義

public *Selection* Selection { get; }

プロパティ値

Selection クラス への参照です。

概要

Selection オブジェクトへの参照を返します。Selection には、現在選択されているアイテムやピースへの参照がコレクションされています。

SizeMode プロパティ

タイムビュー コントロールの表示サイズの決め方を取得または設定します。

定義

public *SizeMode* 列挙型 SizeMode { get; set; }

プロパティ値

SizeMode 列挙型 値のいずれか 1 つ。 既定値は Manual です。

SpecialTimes プロパティ

タイムビュー コントロールの特別時間帯のコレクションを取得します。

定義

public *SpecialTimes* SpecialTimes { get; }

プロパティ値

SpecialTimes クラス への参照です。

概要

特別時間帯を用いて特定の日時の範囲を特別な色で塗りつぶしたりできます。

SumPane プロパティ

タイムビュー コントロールの集計表示ペインを取得します。

定義

public *SumPane* SumPane { get; }

プロパティ値

SumPane クラス への参照です。

概要

集計表示ペインは、設定することによりタイム ビュー コントロールの最上部アイテムの上に表示します。マイルトーンの表示や日付ごとの集計値を表示したりする場所として有用です。

6.1.7 メソッド

GetPieceByKey メソッド

先頭のアイテムから順に、指定の値と同じキー値を持つピースを探索し、最初に見つかったピースを返します。

定義

public *Piece* GetPieceByKey(string key)

パラメーター

key String
取得したいピースのキー値。

戻り値

見つかった *Piece* クラス。

例

次の例は、入力された pieceKey の文字列と Key 値が一致するピースを取得します。

リスト 1: ASP.NET C#

```
protected void GetPieceByKeyButton_Click(
    object sender, EventArgs e)
{
    var pieceKey = Piece_Key.Text;
    var piece = KnTView1.GetPieceByKey(pieceKey);

    ShowPieceInfo(piece);
}
```

GetPieceByValue メソッド

先頭のアイテムから順に、指定の値と同じピース値を持つピースを探索し、最初に見つかったピースを返します。

定義

public *Piece* GetPieceByValue(string key)

パラメーター

key String
取得したいピースの:ref:*Value* <*Piece.Value*>。

戻り値

見つかった ref:*Piece*。

例

次の例は、入力された pieceValue の文字列と Value 値が一致するピースを取得します。

リスト 2: ASP.NET C#

```
protected void GetPieceByValueButton_Click(
    object sender, EventArgs e)
{
    var pieceValue = Piece_Value.Text;
    var piece = KnTView1.GetPieceByValue(pieceValue);

    ShowPieceInfo(piece);
}
```

6.1.8 イベント

AfterPieceAdd イベント

ピースが選択されたことを通知します。

定義

public event AfterPieceAddEventHandler AfterPieceAdd

イベントの種類

AfterPieceAddEventHandler

sender Object
タイムビュー コントロール。

e AfterPieceAddEventArgs
AfterPieceAdd イベント データ。

イベント データ

AfterPieceAddEventArgs

ItemIndex int
ピースが追加されたアイテムのインデックスを取得または設定します。

Start DateTime
追加されたピースの開始日時を取得または設定します。

Finish DateTime
追加されたピースの終了日時を取得または設定します。

Text string
ピースの Text プロパティへ設定して表示したい値を設定します。

Cancel bool
ピースの追加をキャンセルする場合は true を設定します。

例

次の例は、ピースの追加 UI を受け入れ、Text に値を設定します。

リスト 3: ASP.NET C#

```
protected void KnTView1_AfterPieceAdd(
    object sender, AfterPieceAddEventArgs e)
{
    e.Text = "ピース 1";
}
```

AfterPieceSelect イベント

ピースが選択されことを通知します。

定義

public event AfterPieceSelectEventHandler AfterPieceSelect

イベントの種類

AfterPieceSelectEventHandler

sender Object
タイムビュー コントロール。

e AfterPieceSelectEventArgs
AfterPieceSelect イベント データ。

イベント データ

AfterPieceSelectEventArgs

Piece Piece
選択されたピースを表します。ピースの選択が無選択になった場合、この値を null で通知します。

例

次の例は、ピースが選択されたときに、選択されたピースの Index 値をラベルコントロールに表示します。

リスト 4: ASP.NET C#

```
protected void KnTView1_AfterPieceSelect(
    object sender, AfterPieceSelectEventArgs e)
{
    if (e.Piece != null) {
        PieceIndexLabel.Text = e.Piece.Index.ToString();
    } else {
        PieceIndexLabel.Text = string.Empty;
    }
}
```

リスト 5: jQuery

```
afterpieceselect: function (event, piece) {

    if (piece instanceof $.kn.timeview.piece) {

        $('#PieceIndexLabel').text(piece.index());
    } else {
        $('#PieceIndexLabel').text('');
    }
}
```

AfterPieceTimeChange イベント

ユーザー操作によってピースの開始日時または終了日時が変更されたことを通知します。

定義

public event AfterPieceTimeChangeEventHandler AfterPieceTimeChange

イベントの種類

AfterPieceTimeChangeEventHandler

sender Object
タイムビュー コントロール。

e AfterPieceTimeChangeEventArgs
AfterPieceTimeChange イベント データ。

イベント データ

AfterPieceTimeChangeEventArgs

Piece Piece
操作が行われたピースを取得します。

NewStart DateTime
ユーザー操作後の開始日時を取得または設定します。

NewFinish DateTime
ユーザー操作後の終了日時を取得または設定します。

Cancel bool
ユーザー操作をキャンセルする場合は true を設定します。

例

次の例は、ピースの開始日時から終了日時までの長さが 2 日間よりも短くなる操作の場合はキャンセルします。

リスト 6: ASP.NET C#

```
protected void KnTView1_AfterPieceTimeChange(
    object sender, AfterPieceTimeChangeEventArgs e)
{
    if (e.NewFinish - e.NewStart < TimeSpan.FromDays(2))
    {
        e.Cancel = true;
    }
}
```

AfterProgressChange イベント

ユーザー操作によってピースの達成率が変更されたことを通知します。

定義

public event AfterProgressChangeEventHandler AfterProgressChange

イベントの種類

AfterProgressChangeEventHandler

sender Object
タイムビュー コントロール。

e AfterProgressChangeEventArgs
AfterProgressChange イベント データ。

イベント データ

AfterProgressChangeEventArgs

Piece Piece
操作が行われたピースを取得します。

Cancel bool
ユーザー操作をキャンセルする場合は true を設定します。

例
次の例は、ピースの達成率が 0.6 以上なら、ラベルコントロールに"合格"を表示します。

リスト 7: ASP.NET C#

```
protected void KnTView1_AfterProgressChange(
    object sender, AfterProgressChangeEventArgs e)
{
    Piece_Progress.Text = e.Piece.Progress.Value.ToString();

    if (e.Piece.Progress.Value >= 0.6)
    {
        Piece_ProgressStatus.Text = "合格";
    }
    else
    {
        Piece_ProgressStatus.Text = "";
    }
}
```

PieceDblClick イベント

ピースがダブルクリックされことを通知します。

定義
public event PieceDblClickEventHandler PieceDblClick

イベントの種類

PieceDblClickEventHandler

sender Object
タイムビュー コントロール。

e PieceDblClickEventArgs
PieceDblClick イベント データ。

イベント データ

PieceDblClickEventArgs

Piece Piece
ダブルクリックされたピースを取得します。

例
次の例は、ピースがダブルクリックされたら、ピースの背景色を赤色に変更します。

リスト 8: ASP.NET C#

```
protected void KnTView1_PieceDblClick(
    object sender, PieceDblClickEventArgs e)
{
    e.Piece.BackColor = Color.Red;
}
```

6.2 Cells クラス

Cell クラス のコレクションを表します。

6.2.1 概要

行アイテムの *Cells* プロパティから参照します。

6.2.2 プロパティ一覧

プロパティ	内容
Count	セル数を取得します。
Item[int]	指定したインデックスにあるセルを取得または設定します。

6.2.3 プロパティ

Count プロパティ

セルの数を取得します。

定義

public int Count { get; }

プロパティ値

int

例

次の例は、現在のセル数をテキストボックスに表示します。

リスト 9: ASP.NET C#

```
var cells = KnTView1.Items[0].Cells;

Cells_Count.Text = cells.Count.ToString();
```

リスト 10: jQuery

```
var cells = tview.items().item(0).cells();

$("#CellsCount").text(cells.count());
```

Item プロパティ

指定したインデックスにあるセルを取得または設定します。

定義

public *Cell* クラス this[int index] { get; set; }

パラメーター

index int

取得または設定するセルの 0 から始まるインデックス番号。

プロパティ値

Cell クラス

例

次の例は、先頭アイテムの 2 列目のセルを取得して、セルの Text をテキストボックスに表示します。

リスト 11: ASP.NET C#

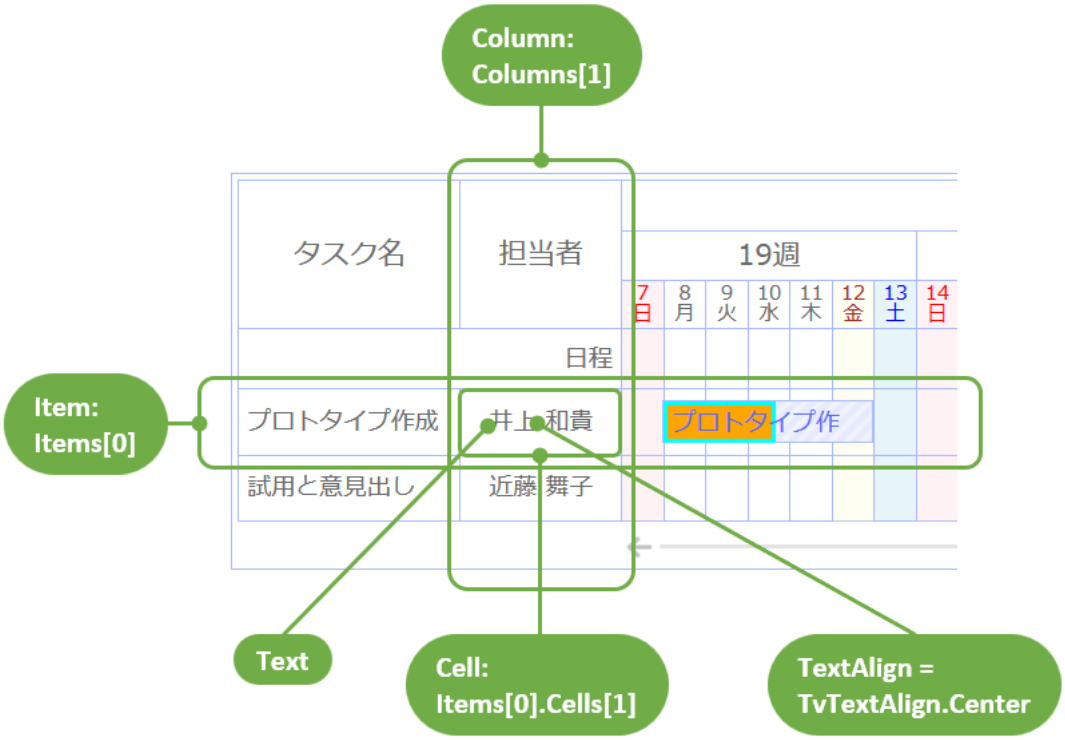
```
var cell = KnTView1.Items[0].Cells[1];

Cell_Text.Text = cell.Text;
```

リスト 12: jQuery

```
var cell = tview.items().item(0).cells().item(1);

CellText.value = cell.text();
```

6.3 Cell クラス

アイテム行と列見出しからなるグリッド上の各マス（セル）を表します。



6.3.1 概要

セルごとに任意の文字列を表示します。

6.3.2 コンストラクター覧

コンストラクタ	内容
Cell()	Cell クラスの新しいインスタンスを初期化します。
Cell(String)	Cell クラスの新しいインスタンスの Text プロパティへ指定の文字列を設定して初期化します。

6.3.3 プロパティ一覧

プロパティ	内容
Text	セルに設定及び表示する文字列を示します。
TextAlign	セルに表示する文字列の横方向の配置を示します。

6.3.4 コンストラクター

Cell クラスの新しいインスタンスを初期化します。

Cell()

Cell クラスの新しいインスタンスを初期化します。

定義

public void Cell();

例

次の例は、既定のコンストラクタにてセルを作成するとともに Text プロパティへ初期値を設定したセルを追加します。

リスト 13: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        var item = new Item();

        var cell = new Cell() { Text = person };
        item.Cells.Add(cell);
    }
}
```

Cell(String)

Cell クラスの新しいインスタンスの Text プロパティへ指定の文字列を設定して初期化します。

パラメーター

Text String
セルに表示する文字列。

例

次の例は、コンストラクタの引数として Text の初期値を与えて作成したセルを追加します。

リスト 14: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        var item = new Item();

        item.Cells.Add(new Cell("カタログ作成"));

        KnTView1.Items.Add(item);
    }
}
```

6.3.5 プロパティ

Text プロパティ

セルに表示する文字列を設定します。値の取得も可能です。

定義

public string Text { get; set; }

プロパティ値

セルに表示する文字列。 既定値は Empty です。

例

次の例は、1 行目 1 列目のセルに「カタログ作成」と表示する方法を示しています。

リスト 15: ASP.NET C#

```
KnTView1.Items[0].Cells[0].Text = "カタログ作成";
```

リスト 16: jQuery

```
tview.items().item(0).cells().item(0).text("カタログ作成");
```

TextAlign プロパティ

セルに表示する文字列の横方向の配置を設定します。値の取得も可能です。

定義

public *TvTextAlign* **列挙型** TextAlign { get; set; }

プロパティ値

TvTextAlign **列挙型** 値のいずれか 1 つ。 既定値は Left です。

例

次の例は、1 行目 2 列目のセルを中央寄せに表示する方法を示しています。

リスト 17: ASP.NET C#

```
KnTView1.Items[0].Cells[1].Align = TvTextAlignCenter;
```

リスト 18: jQuery

```
tview.items().item(0).cells().item(1).textAlign('center');
```

6.4 Columns クラス

列見出しである *Column* **クラス** のコレクションを表します。

6.4.1 概要

タイムビュー コントロールの *Columns* プロパティから参照します。

6.4.2 プロパティ一覧

プロパティ	内容
Count	列見出し数を取得します。
Item[int]	指定したインデックスにある列見出しを取得または設定します。

6.4.3 メソッド一覧

プロパティ	内容
Add	列見出しコレクションに列見出し（列）を追加します。
Clear	列見出しコレクションからすべての列見出し（列）を削除します。
Insert	指定したインデックスの位置に列見出し（列）を挿入します。
Remove	指定の列見出し（列）を列見出しコレクションから削除します。
RemoveAt	指定したインデックスの位置にある列見出し (列) を列見出しコレクションから削除します。

6.4.4 プロパティ

Count プロパティ

列見出し (列) の数を取得します。

定義

public int Count { get; }

プロパティ値

int

例

次の例は、現在の列見出し数をテキストボックスに表示します。

リスト 19: ASP.NET C#

```
Columns_Count.Text = KnTView1.Columns.Count.ToString();
```

リスト 20: jQuery

```
$("#ColumnsCount").text(tview.columns().count());
```

Item プロパティ

指定したインデックスにある列見出し (列) を取得または設定します。

定義

public *Column* クラス this[int index] { get; set; }

パラメーター

index int

取得または設定する列見出しの 0 から始まるインデックス番号。

プロパティ値

Column クラス

例

次の例は、インデックスが 1 番（つまり表示上で左から 2 番目）の列見出しを取得して、Text をテキストボックスに表示します。

リスト 21: ASP.NET C#

```
var column = KnTView1.Columns[1].Text;

Column_Text.Text = column.Text;
```

リスト 22: jQuery

```
var column = tview.columns().item(1);

ColumnText.value = column.text();
```

6.4.5 メソッド

Add(Column)

列見出しコレクションに指定の列見出しを末尾に追加します。

定義

public void Add(*Column* column);

パラメーター

column ***Column*** **クラス**

追加する *Column* クラス。

例

次の例は、80 ピクセルの幅で「タスク名」というタイトルの列見出しを追加します。

リスト 23: ASP.NET C#

```
using Knowlbo.Component.Web.TimeView;
using System;

namespace KnTViewSample1
{
    public partial class ColumnsText : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
```

(次のページに続く)

(前のページからの続き)

```
            {
                KnTView1.Columns.Add(new Column("タスク名", 80));
            }
        }
    }
}
```

リスト 24: jQuery

```
$tview.timeview("addColumn", {text: "タスク名", width: 80});
```

Clear()

列見出しコレクションからすべての列見出し（列）を削除します。

定義

public void Clear();

例

次の例は、3 列追加した後に、この追加した列をすべて削除します。

リスト 25: ASP.NET C#

```
using Knowlbo.Component.Web.TimeView;
using System;

namespace KnTViewSample1
{
    public partial class ColumnsText : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                KnTView1.Columns.Add(new Column("タスク名", 80));
                KnTView1.Columns.Add(new Column("担当者", 70));
                KnTView1.Columns.Add(new Column("状況", 50));

                KnTView1.Columns.Clear();
            }
        }
    }
}
```

Insert(int, Column)

指定したインデックスの位置に列見出し（列）を挿入します。

定義

public void Insert(int index, *Column* column);

パラメーター

index int
列見出しを挿入する位置の、0 から始まるインデックス。

column *Column* クラス
追加する *Column* クラス。

例

次の例は、インデックスが 1 番（つまり表示上で左から 2 番目）の位置に列見出しを挿入します。

リスト 26: ASP.NET C#

```
var newColumn = new Column("担当者", 70);

KnTView1.Columns.Insert(1, newColumn);
```

Remove(Column)

列見出しコレクション内に存在する指定の列見出しを削除します。

定義

public void Remove(*Column* column);

パラメーター

column *Column* クラス
削除する *Column* クラス。

例

次の例は、追加した列見出しと同じ列見出しを削除します。

リスト 27: ASP.NET C#

```
using Knowlbo.Component.Web.TimeView;
using System;

namespace KnTViewSample1
{
    public partial class ColumnsText : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
```

(次のページに続く)

(前のページからの続き)

```
            if (!IsPostBack)
            {
                var taskNameColumn = new Column("タスク名", 80);

                KnTView1.Columns.Add(taskNameColumn);
                KnTView1.Columns.Remove(taskNameColumn);
            }
        }
    }
}
```

RemoveAt(int)

指定したインデックスの位置にある列見出し（列）を列見出しコレクションから削除します。

定義

public void RemoveAt(int index);

パラメーター

index int
列見出しを削除する位置の、0 から始まるインデックス。

例

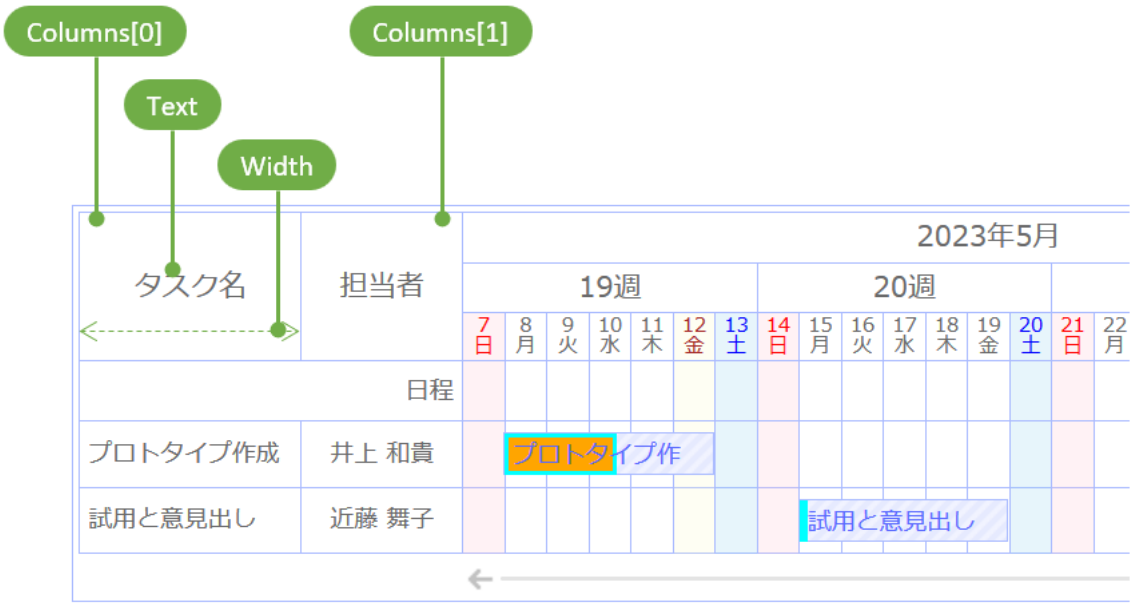
次の例は、インデックスが 1 番（つまり表示上で左から 2 番目）の位置にある列見出しを削除します。

リスト 28: ASP.NET C#

```
KnTView1.Columns.RemoveAt(1);
```

6.5 Column クラス

列の列見出しを表します。



6.5.1 概要

Column クラスは、列の見出し部分を表します。Text プロパティに設定した文字列が、列見出しに表示されます。Width プロパティを使って、列の幅を制御することができます。

6.5.2 コンストラクター覧

コンストラクタ	内容
Column()	Column クラスの新しいインスタンスを初期化します。
Column(String, int)	Column クラスの新しいインスタンスの Text プロパティおよび Width プロパティへ指定の値を設定して初期化します。

6.5.3 プロパティ一覧

プロパティ	内容
Text	列見出しに表示する文字列を取得または設定します。
Width	列見出しの幅を取得または設定します。

6.5.4 コンストラクター

Column クラスの新しいインスタンスを初期化します。

Column()

Column クラスの新しいインスタンスを初期値で作成します。

定義

public void Column();

例

次の例は、既定のコンストラクタにて作成するとともに Text および Width プロパティへ初期値を設定した列見出しを追加します。

リスト 29: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Columns.Add(
            new Column() { Text = "タスク名", Width = 110 });
    }
}
```

Column(String, int)

Column クラスの新しいインスタンスの Text プロパティおよび Width プロパティへ指定の値を設定して初期化します。

パラメーター

text String

列見出しに表示するタイトル文字列。

width int

列見出しの幅 (ピクセル)。

例

次の例は、コンストラクタの引数として Text および Width の初期値を与えて作成した列見出しを追加します。

リスト 30: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Columns.Add(new Column("担当者", 80));
    }
}
```

6.5.5 プロパティ

Text プロパティ

列見出しに表示するタイトル文字列を設定します。値の取得も可能です。

定義

public string Text { get; set; }

プロパティ値

列見出しに表示する文字列。 既定値は Empty です。

例

次の例は、列見出しのタイトルを「作業名」に変更する方法を示しています。

リスト 31: ASP.NET C#

```
KnTView1.Columns[0].Text = "作業名";
```

リスト 32: jQuery

```
tview.columns().item(0).text("作業名");
```

Width プロパティ

列見出しの幅をピクセル数で設定します。値の取得も可能です。

定義

public int Width { get; set; }

プロパティ値

列見出しの幅を表す数値。単位はピクセル。既定値は 0 です。

例

次の例は、列見出しの幅を 100 ピクセルに設定する方法を示しています。

リスト 33: ASP.NET C#

```
KnTView1.Columns[0].Width = 100;
```

リスト 34: jQuery

```
tview.columns().item(0).width(100);
```

6.6 Items クラス

行アイテムである *Item クラス* のコレクションを表します。

6.6.1 概要

タイムビュー コントロールの *Items* プロパティから参照します。

6.6.2 プロパティ一覧

プロパティ	内容
Count	行アイテム数（行数）を取得します。
Item[int]	指定したインデックスにある行アイテムを取得または設定します。

6.6.3 メソッド一覧

プロパティ	内容
Add	行アイテムコレクションに行アイテム（行）を追加します。
Clear	行アイテムコレクションからすべての行アイテム（行）を削除します。
Insert	指定したインデックスの位置に行アイテム（行）を挿入します。
Remove	指定の行アイテム（行）を行アイテムコレクションから削除します。
RemoveAt	指定したインデックスの位置にある行アイテム（行）を行アイテムコレクションから削除します。

6.6.4 プロパティ

Count プロパティ

行アイテム数 (行数) を取得します。

定義

public int Count { get; }

プロパティ値

int

例

次の例は、現在の行アイテム数をテキストボックスに表示します。

リスト 35: ASP.NET C#

```
Items_Count.Text = KnTView1.Items.Count.ToString();
```

リスト 36: jQuery

```
$("#ItemsCount").text(tview.items().count());
```

Item プロパティ

指定したインデックスにある行アイテム (行) を取得または設定します。

定義

public *Item* クラス this[int index] { get; set; }

パラメーター

index int

取得または設定する行アイテムの 0 から始まるインデックス番号。

プロパティ値

Item クラス

例

次の例は、インデックスが 1 番（つまり表示上で上から 2 番目）の行アイテムを取得して、1 列目のセルの Text をテキストボックスに表示します。

リスト 37: ASP.NET C#

```
var item = KnTView1.Items[1];

Cell_Text.Text = item.Cells[0].Text;
```

リスト 38: jQuery

```
var item = tview.items().item(1);

CellText.value = item.cells().item(0).text();
```

6.6.5 メソッド

Add(Item)

行アイテムコレクションに指定の行アイテムを末尾に追加します。

定義

public void Add(*Item* item);

パラメーター

item ***Item* クラス**
追加する *Item* クラス。

例

次の例は、1 列目のセルに「アイテム 1」という文字列を表示する行アイテムを追加します。

リスト 39: ASP.NET C#

```
using Knowlbo.Component.Web.TimeView;
using System;

namespace KnTViewSample1
{
    public partial class ItemsText : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                KnTView1.Items.Add(new Item("アイテム 1"));
            }
        }
    }
}
```

リスト 40: jQuery

```
$tview.timeview("addItem", { text: "アイテム 1" });
```

Clear()

行アイテムコレクションからすべての行アイテム（行）を削除します。

定義

public void Clear();

例

次の例は、3 行追加した後に、行をすべて削除します。

リスト 41: ASP.NET C#

```
using Knowlbo.Component.Web.TimeView;
using System;

namespace KnTViewSample1
{
    public partial class ItemsText : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                KnTView1.Items.Add(new Item("執筆"));
                KnTView1.Items.Add(new Item("安田"));
                KnTView1.Items.Add(new Item("30"));

                KnTView1.Items.Clear();
            }
        }
    }
}
```

Insert(int, Item)

指定したインデックスの位置に行アイテム（行）を挿入します。

定義

public void Insert(int index, *Item* item);

パラメーター

index int
行アイテムを挿入する位置の、0 から始まるインデックス。

item *Item* クラス
追加する *Item* クラス。

例

次の例は、インデックスが 1 番（つまり表示上で上から 2 番目）の位置に行アイテムを挿入します。

リスト 42: ASP.NET C#

```
var newItem = new Item("推敲");

KnTView1.Items.Insert(1, newItem);
```

Remove(Item)

行アイテムコレクション内に存在する指定の行アイテムを削除します。

定義

public void Remove(*Item* item);

パラメーター

item *Item* クラス
削除する *Item* クラス。

例

次の例は、追加した行アイテムと同じ行アイテムを削除します。

リスト 43: ASP.NET C#

```
using Knowlbo.Component.Web.TimeView;
using System;

namespace KnTViewSample1
{
    public partial class ItemsText : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                var item = new Item("執筆");

                KnTView1.Items.Add(item);
                KnTView1.Items.Remove(item);
            }
        }
    }
}
```

RemoveAt(int)

指定したインデックスの位置にある行アイテム（行）を行アイテムコレクションから削除します。

定義

public void RemoveAt(int index);

パラメーター

index int

行アイテムを削除する位置の、0 から始まるインデックス。

例

次の例は、インデックスが 1 番（つまり表示上で上から 2 番目）の位置にある行アイテムを削除します。

リスト 44: ASP.NET C#

```
KnTView1.Items.RemoveAt(1);
```

6.7 Item クラス

行を表します。

6.7.1 概要

Item クラスは、タイムビューの行を表します。

6.7.2 コンストラクター一覧

コンストラクタ	内容
Item()	新しい行アイテムを既定のプロパティ値で初期化して作成します。
Item(String)	新しい行アイテムを 1 列目のセルに表示する文字列を設定して作成します。

6.7.3 プロパティ一覧

プロパティ	内容
Cells	アイテムが持つセルのコレクションを取得します。
Folded	折り畳みによって下位レベルのアイテムを隠すかどうかを取得または設定します。
Index	アイテム コレクション内での位置を表すインデックスを取得します。
Level	アイテムのインデント レベルを取得または設定します。
Pieces	アイテムが持つピースのコレクションを取得します。

6.7.4 コンストラクター

Item クラスの新しいインスタンスを初期化します。

Item()

Item クラスの新しいインスタンスを初期値で作成します。

定義

public void Item();

例

次の例は、既定のコンストラクタにて行を作成して末尾に追加します。

リスト 45: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Items.Add(new Item());
    }
}
```

Item(String)

Item クラスの新しいインスタンスの 1 列目のセルに表示する文字列を設定して初期化します。

パラメーター

cellText String
1 列目のセルに表示する文字列。

例

次の例は、1 列目のセルに"執筆"という文字列を表示するアイテムを末尾に追加します。

リスト 46: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Items.Add(new Item("執筆"));
    }
}
```

6.7.5 プロパティ

Cells プロパティ

このアイテムが持つセルのコレクションを取得します。

定義

public string Cells { get; set; }

プロパティ値

Cells クラス への参照です。

概要

セル数は列見出し数と一致します。セルのインデックス値も列見出しのインデックス値と一致します。

例

次の例は、列見出しを 2 列追加したあとにアイテムを追加して 2 列目のセルにテキストを表示します。セル数が列見出し数と一致するのはアイテムがアイテム コレクションへ追加した時点です。個別のセル オブジェクトへのアクセスは、アイテムをアイテム コレクションへ追加したあとに行ってください。

リスト 47: ASP.NET C#

```
KnTView1.Columns.Add(new Column("タスク名", 100));
KnTView1.Columns.Add(new Column("作業者", 80));

var newItem = new Item();
KnTView1.Items.Add(newItem);

newItem.Cells[0].Text = "執筆";
newItem.Cells[1].Text = "井上 和貴";
```

リスト 48: jQuery

```
$tview = $('#kntview1').timeview({
    columns: [
```

(次のページに続く)

(前のページからの続き)

```
        { text: "タスク名", width: 100 },
        { text: "作業者", width: 80 }
    ]
});

var newItem = $tview.timeview("addItem");

newItem.cells().item(0).text("執筆");
newItem.cells().item(0).text("井上 和貴");
```

Folded プロパティ

下位レベルのアイテムを折り畳む（非表示にする）かどうかを取得または設定します。レベルは数値が小さいほど上位となり、数値が大きいほど解であると扱います。折りたたむ範囲は、このアイテムとレベルが同一または上位レベルのアイテムが現れる手前までです。TODO: ASP.NET 版は子の仕様通り動作しない。修正するかマニュアルと仕様から消すかが迫られる。

定義

public bool Folded { get; set; }

プロパティ値

列見出しに表示する文字列。 既定値は Empty です。

例

次の例は、1 行目よりも下位レベルで連続するアイテムを折りたたみます（非表示にします）

リスト 49: ASP.NET C#

```
KnTView1.Items[0].Folded = true;
```

リスト 50: jQuery

```
tview.items().item(0).folded(true);
```

Index プロパティ

アイテム コレクション内での位置を表す一番上の行を 0 としたインデックスを取得します。

定義

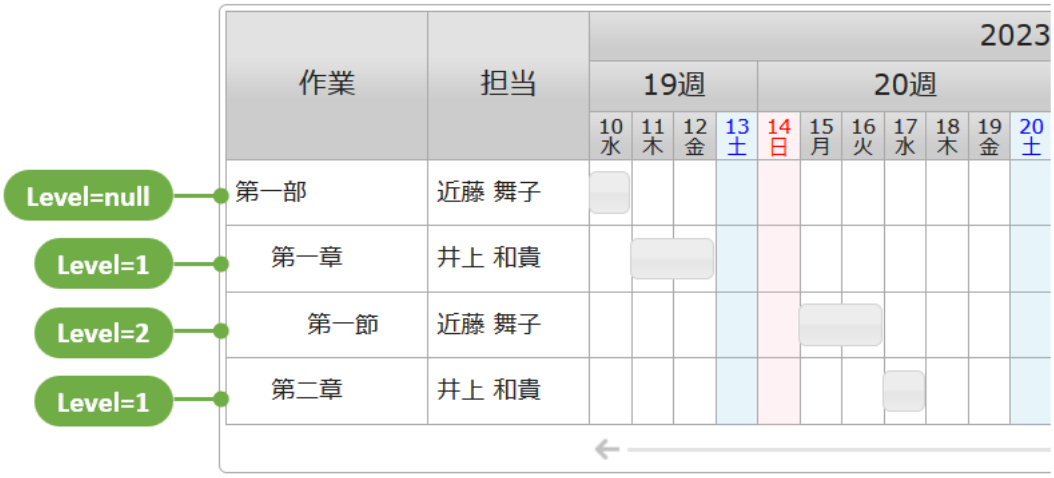
public int Index { get; }

プロパティ値

アイテム コレクション内でアイテムが位置する 0 から始まるインデックス番号。

Level プロパティ

アイテムのインデントの深さを示すレベル値を取得または設定します。1 つ上のアイテムよりも大きい数字を設定するとテキストを右に字下げして表示します。



定義

public int? Level { get; set; }

プロパティ値

インデント レベルを表す 0 以上の数値。既定値は null (設定なし) です。

例

次の例は、レベルを設定して章立ての階層を表現します。

リスト 51: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Columns.Add(new Column("作業", 100));
        KnTView1.Columns.Add(new Column("担当", 80));

        var newItem = new Item();
        KnTView1.Items.Add(newItem);
        newItem.Cells[0].Text = "第一部";
        newItem.Cells[1].Text = "近藤 舞子";

        newItem = new Item();
        KnTView1.Items.Add(newItem);
        newItem.Level = 1;
        newItem.Cells[0].Text = "第一章";
        newItem.Cells[1].Text = "井上 和貴";
```

(次のページに続く)

(前のページからの続き)

```
        newItem = new Item();
        KnTView1.Items.Add(newItem);
        newItem.Level = 2;
        newItem.Cells[0].Text = "第一節";
        newItem.Cells[1].Text = "近藤 舞子";

        newItem = new Item();
        KnTView1.Items.Add(newItem);
        newItem.Level = 1;
        newItem.Cells[0].Text = "第二章";
        newItem.Cells[1].Text = "井上 和貴";
    }
}
```

リスト 52: jQuery

```
$tview = $('#kntview1').timeview({
    columns: [
        { text: "作業", width: 100 },
        { text: "担当", width: 80 }
    ]
});

var newItem = $tview.timeview("addItem");
newItem.cells().item(0).text("第一部");
newItem.cells().item(1).text("近藤 舞子");

newItem = $tview.timeview("addItem");
newItem.level(1);
newItem.cells().item(0).text("第一章");
newItem.cells().item(1).text("井上 和貴");

newItem = $tview.timeview("addItem");
newItem.level(2);
newItem.cells().item(0).text("第一節");
newItem.cells().item(1).text("近藤 舞子");

newItem = $tview.timeview("addItem");
newItem.level(1);
newItem.cells().item(0).text("第二章");
newItem.cells().item(1).text("井上 和貴");

$tview.timeview("option", "sizeMode", "autoHeight");
```

Pieces プロパティ

アイテムが持つピースのコレクションを取得します。

定義

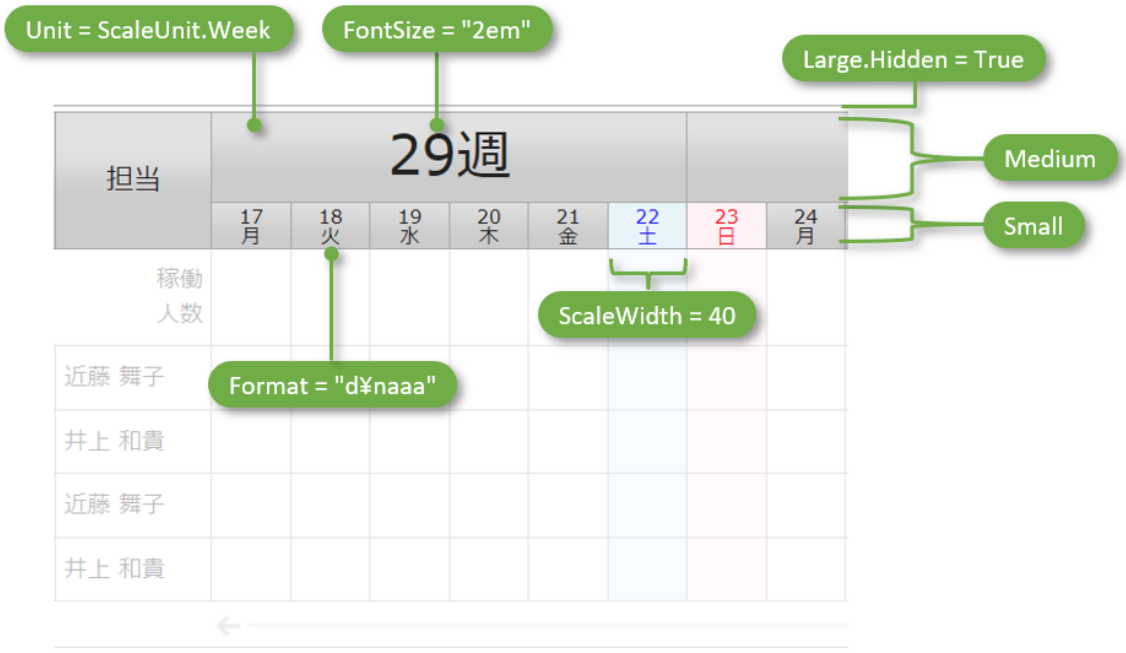
public *Pieces* Pieces { get; }

プロパティ値

Pieces クラス への参照です。

6.8 LetterRuler クラス

タイム ルーラーの各区分（大、中、小）の目盛りを表します。



6.8.1 概要

タイム ルーラーの各区分の目盛りの単位や表記を設定できます。

6.8.2 プロパティ一覧

プロパティ	内容
FontSize	目盛りのフォントサイズを取得または設定します。
Format	目盛りの表示書式を取得または設定します。
Hidden	当区分の目盛りを非表示にするかどうかを取得または設定します。
ScaleWidth	目盛り同士の間隔を取得または設定します。
Unit	目盛りの単位を取得または設定します。

6.8.3 プロパティ

FontSize プロパティ

目盛りのフォントサイズ（表示する文字の大きさ）を取得または設定します。

定義

public string FontSize { get; set; }

プロパティ値

目盛りのフォントサイズを CSS の font-size に設定する形式で文字列にて指定します。

例

次の例は、中区分の目盛りを 2em （既定の文字サイズの 2 倍の大きさ）で表示します。

リスト 53: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Ruler.Medium.FontSize = "2em";
    }
}
```

リスト 54: jQuery

```
const $tview = $('#kntview1').timeview({
    timeRuler: {
        medium: {
            fontSize: "2em"
        }
    }
});
```

Format プロパティ

目盛りの表示書式を取得または設定します。

定義

public string Format { get; set; }

プロパティ値

目盛りの日付表示書式を指定します。

日付表示書式

書式	内容
yyyy	西暦の年を 4 桁の数値で返します (100 ～ 9999)。
m	月を表す数値を返します。1 桁の場合、先頭に 0 が付きません (1 ～ 12)。ただし、h や hh の直後に m を指定した場合、月ではなく分と解釈されます。
ww	その日が一年のうちで何週目に当たるかを表す数値を返します (1 ～ 54)。
d	日付を返します。1 桁の場合、先頭に 0 が付きません (1 ～ 31)。
aaa	曜日を日本語 (省略形) で返します (日～土)。
n	改行します。

例

次の例は、小区分の目盛りについて日付と曜日を改行を用いて縦に表示します。

リスト 55: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Ruler.Small.Format = "d\naaa";
    }
}
```

リスト 56: jQuery

```
const $tview = $('#kntview1').timeview({
    timeRuler: {
        small: {
            format: "d\naaa"
        }
    }
});
```

Hidden プロパティ

当区分の目盛りを非表示にするかどうかを取得または設定します。

定義

public bool Hidden { get; set; }

プロパティ値

当該区分の目盛りを非表示にする場合は true 、表示する場合は false を設定します。

小区分は非表示にすることができません。

例

次の例は、大区分を非表示にして、中区分と小区分のみ表示します。

リスト 57: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Ruler.Large.Hidden = true;
    }
}
```

リスト 58: jQuery

```
const $tview = $('#kntview1').timeview({
    timeRuler: {
        large: {
            hidden: true
        }
    }
});
```

ScaleWidth プロパティ

目盛り幅をピクセル数で設定、または取得します。このプロパティは小区分でのみ有効です。

定義

public int? ScaleWidth { get; set; }

プロパティ値

1 以上の数値を指定します。

例

次の例では、40 ピクセル幅で表示します。

リスト 59: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Ruler.Small.ScaleWidth = 40;
    }
}
```

リスト 60: jQuery

```
const $tview = $('#kntview1').timeview({
    timeRuler: {
        small: {
            scaleWidth: 40
        }
    }
});
```

Unit プロパティ

目盛りの単位を取得または設定します。

定義

public *ScaleUnit* 列挙型 Unit { get; set; }

プロパティ値

ScaleUnit 列挙型 値のいずれか 1 つ。 既定値は Left です。

例

次の例は、中区分を 1 週間単位の目盛りにします。

リスト 61: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Ruler.Medium.Unit = ScaleUnit.Week;
        KnTView1.Ruler.Medium.Format = "m 月 d 日の週";
    }
}
```

リスト 62: jQuery

```
const $tview = $('#kntview1').timeview({
    timeRuler: {
        medium: {
            unit: "week",
            format: "m 月 d 日の週"
        },
    }
});
```

6.9 Pieces クラス

ピースである *Piece* クラス のコレクションを表します。

6.9.1 概要

Item クラス の *Pieces* プロパティから参照します。

6.9.2 プロパティ一覧

プロパティ	内容
Count	ピース数を取得します。
Item[int]	指定したインデックスにあるピースを取得または設定します。

6.9.3 メソッド一覧

プロパティ	内容
Add	ピースコレクションにピース（列）を追加します。
Clear	ピースコレクションからすべてのピース（列）を削除します。
Insert	指定したインデックスの位置にピース（列）を挿入します。
Remove	指定のピース（列）をピースコレクションから削除します。
RemoveAt	指定したインデックスの位置にあるピース（列）をピースコレクションから削除します。

6.9.4 プロパティ

Count プロパティ

ピース (列) の数を取得します。

定義

public int Count { get; }

プロパティ値

int

例

次の例は、現在のピース数をテキストボックスに表示します。

リスト 63: ASP.NET C#

```
Pieces_Count.Text = KnTView1.Pieces.Count.ToString();
```

リスト 64: jQuery

```
$("#PiecesCount").text(tview.columns().count());
```

Item プロパティ

指定したインデックスにあるピース (列) を取得または設定します。

定義

public *Piece* クラス this[int index] { get; set; }

パラメーター

index int

取得または設定するピースの 0 から始まるインデックス番号。

プロパティ値

Piece クラス

例

次の例は、インデックスが 1 番（つまり表示上で左から 2 番目）のピースを取得して、Text をテキストボックスに表示します。

リスト 65: ASP.NET C#

```
var column = KnTView1.Pieces[1].Text;

Piece_Text.Text = column.Text;
```

リスト 66: jQuery

```
var column = tview.columns().item(1);

PieceText.value = column.text();
```

6.9.5 メソッド

Add(Piece)

ピースコレクションに指定のピースを末尾に追加します。

定義

public void Add(*Piece* column);

パラメーター

column *Piece* クラス
追加する *Piece* クラス。

例

次の例は、80 ピクセルの幅で「タスク名」というタイトルのピースを追加します。

リスト 67: ASP.NET C#

```
using Knowlbo.Component.Web.TimeView;
using System;

namespace KnTViewSample1
{
    public partial class PiecesText : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                KnTView1.Pieces.Add(new Piece("タスク名", 80));
            }
        }
    }
}
```

リスト 68: jQuery

```
$tview.timeview("addPiece", { itemIndex: 1, start: "2014/4/18", finish: "2014/4/24",␣
↪caption: "海外研修" });
```

Clear()

ピースコレクションからすべてのピース（列）を削除します。

定義

public void Clear();

例

次の例は、3 列追加した後に、この追加した列をすべて削除します。

リスト 69: ASP.NET C#

```
using Knowlbo.Component.Web.TimeView;
using System;

namespace KnTViewSample1
{
    public partial class PiecesText : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                KnTView1.Pieces.Add(new Piece("タスク名", 80));
                KnTView1.Pieces.Add(new Piece("担当者", 70));
                KnTView1.Pieces.Add(new Piece("状況", 50));

                KnTView1.Pieces.Clear();
            }
        }
    }
}
```

Insert(int, Piece)

指定したインデックスの位置にピース（列）を挿入します。

定義

public void Insert(int index, *Piece* column);

パラメーター

index int
ピースを挿入する位置の、0 から始まるインデックス。

column *Piece* クラス
追加する *Piece* クラス。

例

次の例は、インデックスが 1 番（つまり表示上で左から 2 番目）の位置にピースを挿入します。

リスト 70: ASP.NET C#

```
var newPiece = new Piece("担当者", 70);

KnTView1.Pieces.Insert(1, newPiece);
```

Remove(Piece)

ピースコレクション内に存在する指定のピースを削除します。

定義

public void Remove(*Piece* column);

パラメーター

column *Piece* クラス
削除する *Piece* クラス。

例

次の例は、追加したピースと同じピースを削除します。

リスト 71: ASP.NET C#

```
using Knowlbo.Component.Web.TimeView;
using System;

namespace KnTViewSample1
{
    public partial class PiecesText : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                var taskNamePiece = new Piece("タスク名", 80);

                KnTView1.Pieces.Add(taskNamePiece);
                KnTView1.Pieces.Remove(taskNamePiece);
            }
        }
    }
}
```

RemoveAt(int)

指定したインデックスの位置にあるピース（列）をピースコレクションから削除します。

定義

public void RemoveAt(int index);

パラメーター

index int

ピースを削除する位置の、0 から始まるインデックス。

例

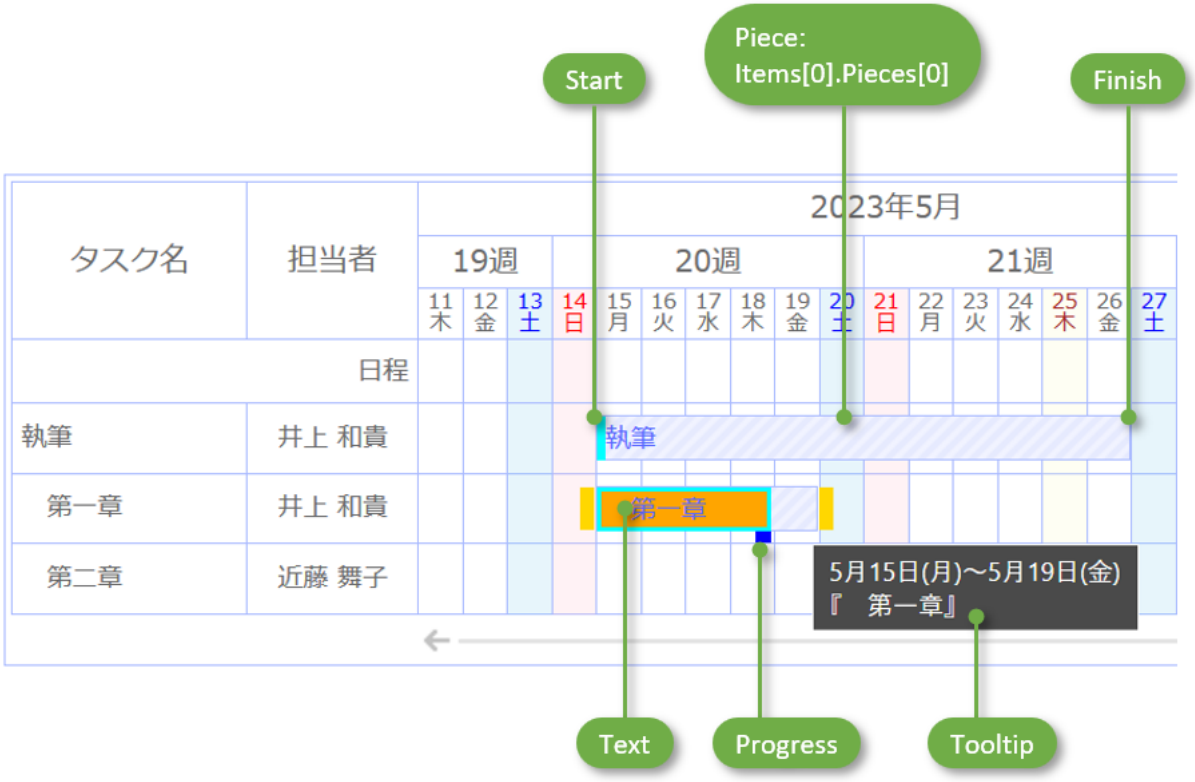
次の例は、インデックスが 1 番（つまり表示上で左から 2 番目）の位置にあるピースを削除します。

リスト 72: ASP.NET C#

```
KnTView1.Pieces.RemoveAt(1);
```

6.10 Piece クラス

各アイテムのタイムルーラー (時間軸) 下に表示するバーをピースと呼びます。



6.10.1 概要

ピースは、作業や予定など、特定の日時や期間に起こる事柄を表現する、基本的な単位を表します。

6.10.2 コンストラクター一覧

コンストラクタ	内容
Piece()	Piece クラスの新しいインスタンスを初期化します。
Piece(DateTime, DateTime, String)	Piece クラスの新しいインスタンスをバーの期間、表示テキストを明示的に設定して初期化します。

6.10.3 プロパティー一覧

プロパティ	内容
BackColor	ピースを塗りつぶす背景色を取得または設定します。
BorderColor	ピースの枠線の色を取得または設定します。
Finish	ピースの終了日時を取得または設定します。
ForeColor	ピースに表示するテキストの色を取得または設定します。
Index	当ピースのピース コレクション内での位置を表すインデックスを取得します。
ItemIndex	当ピースが属するアイテムのアイテム コレクション内での位置を表すインデックスを取得します。
Key	ピース コレクション内で自身を識別する文字列を取得または設定します。
Progress	ピースの達成率を表すオブジェクトを取得します。
Start	ピースの開始日時を取得または設定します。
Text	ピースに表示する文字列を取得または設定します。
ToolTip	ピースにマウスを重ねたときに表示する文字列を取得または設定します。
Value	ピースに保持させたい任意の値を取得または設定します。

6.10.4 コンストラクター

Piece クラスの新しいインスタンスを初期化します。

Piece()

Piece クラスの新しいインスタンスを初期化します。

定義

public void Piece();

例

次の例は、今日から 5 日間で行う「執筆/第一章」という名の作業を表すピースを新しいアイテム行に追加します。

リスト 73: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        var newItem = new Item();
        KnTView1.Items.Add(newItem);

        var newPiece = new Piece()
        {
            Start = DateTime.Today,
            Finish = DateTime.Today.AddDays(5),
            Text = "執筆/第一章"
        };
        newItem.Pieces.Add(newPiece);
    }
}
```

Piece(DateTime, DateTime, String)

Piece クラスの新しいインスタンスをバーの期間、表示テキストを明示的に設定して初期化します。

パラメーター

start DateTime
ピースの開始日時。

finish DateTime
ピースの終了日時。

text String
ピースに表示する文字列。

例

次の例は、今日から 5 日間で行う「執筆/第一章」という名の作業を表すピースを新しいアイテム行に追加します。

リスト 74: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        var newItem = new Item();
        KnTView1.Items.Add(newItem);

        var newPiece = new Piece(
            DateTime.Today,
            DateTime.Today.AddDays(5),
            "執筆/第一章");
        newItem.Pieces.Add(newPiece);
    }
}
```

6.10.5 プロパティ

BackColor プロパティ

ピースを塗りつぶす背景色を取得または設定します。

定義

public Color BackColor { get; set; }

プロパティ値

ピースの背景色を表す Color。既定値は Empty です。既定値の場合は jQuery UI のテーマに沿った背景色を使用します。

例

次の例は、ピンク色のピースを表示します。

リスト 75: ASP.NET C#

```
var newItem = new Item();
KnTView1.Items.Add(newItem);

var newPiece = new Piece()
{
    Start = DateTime.Today,
    Finish = DateTime.Today.AddDays(5),
    Text = "第一章/井上",
    BackColor = Color.Pink
};
newItem.Pieces.Add(newPiece);
```

リスト 76: jQuery

```
var now = new Date();
var today = new Date(now.getFullYear(), now.getMonth(), now.getDate());

var newItem = $tview.timeview("addItem");
newPiece = $tview.timeview("addPiece", {
    itemIndex: newItem.index(),
    start: today,
    finish: new Date(today.getFullYear(), today.getMonth(), today.getDate() + 5),
    caption: "第一章/井上"
});
newPiece.backgroundColor("pink");
```

BorderColor プロパティ

ピースの枠線の色を取得または設定します。

定義

public Color BorderColor { get; set; }

プロパティ値

ピースの枠線の色を表す Color。既定値は Empty です。既定値の場合は jQuery UI のテーマに沿った枠線の色を使用します。

例

次の例は、赤い枠線のピースを表示します。

リスト 77: ASP.NET C#

```
var newItem = new Item();
KnTView1.Items.Add(newItem);

var newPiece = new Piece()
{
    Start = DateTime.Today,
    Finish = DateTime.Today.AddDays(5),
    Text = "第一章/井上",
    BackColor = Color.Pink,
    BorderColor = Color.Red,
};
newItem.Pieces.Add(newPiece);
```

リスト 78: jQuery

```
var now = new Date();
var today = new Date(now.getFullYear(), now.getMonth(), now.getDate());

var newItem = $tview.timeview("addItem");
newPiece = $tview.timeview("addPiece", {
    itemIndex: newItem.index(),
    start: today,
    finish: new Date(today.getFullYear(), today.getMonth(), today.getDate() + 5),
    caption: "第一章/井上"
});
newPiece.backgroundColor("pink");
newPiece.borderColor("red");
```

Finish プロパティ

ピースの終了日時を取得または設定します。

定義

public DateTime Finish { get; set; }

プロパティ値

ピースの終了日時。Finish プロパティの値は、Start プロパティの値以上を設定します。

例

次の例は、選択したピースの終了日時を 1 日延長します。

リスト 79: ASP.NET C#

```
protected void PieceFinishExtendOneDayButton_Click(object sender, EventArgs e)
{
    if (KnTView1.Selection.Pieces.Count > 0)
    {
        var piece = KnTView1.Selection.Pieces[0];

        piece.Finish += TimeSpan.FromDays(1);
    }
}
```

リスト 80: jQuery

```
$("#ExtendOneDayButton").button().click(function () {

    var selection = $tview.timeview("selection");
    if (selection.pieces().count() > 0) {
```

(次のページに続く)

(前のページからの続き)

```
        var piece = selection.pieces().item(0);

        var finish = piece.finish();
        var newFinish = new Date(finish.getFullYear(), finish.getMonth(), finish.
↪getDate() + 1);

        piece.finish(newFinish);
    }
});
```

ForeColor プロパティ

ピースに表示するテキストの色を取得または設定します。

定義

public string ForeColor { get; set; }

プロパティ値

ピースに表示する文字列。 既定値は Empty です。

例

次の例は、1 行目 1 列目のピースに「カタログ作成」と表示する方法を示しています。

リスト 81: ASP.NET C#

```
KnTView1.Items[0].Pieces[0].ForeColor = "カタログ作成";
```

リスト 82: jQuery

```
tview.items().item(0).pieces().item(0).text("カタログ作成");
```

Index プロパティ

当ピースのピース コレクション内での位置を表すインデックスを取得します。

定義

public int Index { get; set; }

プロパティ値

ピースに表示する文字列。 既定値は Empty です。

例

ピース コレクション内で当ピースが位置する 0 から始まるインデックス番号。

ItemIndex プロパティ

当ピースが属するアイテムのアイテム コレクション内での位置を表すインデックスを取得します。

定義

public int ItemIndex { get; set; }

プロパティ値

当ピースを含むアイテムがアイテム コレクション内で位置する 0 から始まるインデックス番号。

Key プロパティ

ピース コレクション内で自身を識別する文字列を取得または設定します。

定義

public string Key { get; set; }

プロパティ値

ピース コレクション内の当ピースを識別する任意の文字列。

例

次の例は、CSV に作成した 宿泊予約一覧の主キー 予約 ID の値をピースの Key に設定して作成し、ピースクリック時に Key をキーに 予約の詳細を取得して表示します。

リスト 83: ASP.NET C#

```
protected void KnTView1_AfterPieceSelect(object sender, AfterPieceSelectEventArgs e)
{
    foreach (var room in _rooms.Values)
    {
        var reserveId = int.Parse(e.Piece.Key);

        if (room.Reserves.ContainsKey(reserveId))
        {
            var reserve = room.Reserves[reserveId];

            ReserveNameLabel.Text = reserve.Name;
            ReserveRoomLabel.Text = reserve.Room;
            ReserveStartDateLabel.Text = reserve.StartDate.ToString("yyyy 年 M 月 d 日␣
↪dddd");
            ReserveStaysLabel.Text = reserve.Stays.ToString("0 泊");
            break;
        }
    }
}
KnTView1.Items[0].Pieces[0].Key = "カタログ作成";
```

リスト 84: jQuery

```
var $tview = $('#kntview1').timeview({
    columns: [
        { text: "部屋番号", width: 100 },
    ],
    afterpieceselect: function (event, piece) {
        if (piece) {

            rooms.forEach((room) => {

                var reserve = room.reserves.find((reserve) =>
                    reserve.reserveId === piece.key()
                );
                if (reserve) {

                    $('#ReserveNameLabel').text(reserve.name);
                    $('#ReserveRoomLabel').text(reserve.room);
                    $('#ReserveStartDateLabel').text('' +
                        reserve.startDate.getFullYear() + '年' +
                        (reserve.startDate.getMonth() + 1) + '月' +
                        reserve.startDate.getDate() + '日');
                    $('#ReserveStaysLabel').text(reserve.stays);
                }
            });
        }
    }
});
```

Progress プロパティ

ピースの達成率を表すオブジェクトを取得します。

定義

public *Progress* Progress { get; }

プロパティ値

Progress クラス への参照です。

例

次の例は、ピース作成時に達成率を 25% に設定します。

リスト 85: ASP.NET C#

```
var newPiece = new Piece()
{
```

(次のページに続く)

(前のページからの続き)

```
    Start = DateTime.Today,
    Finish = DateTime.Today.AddDays(5),
    Text = "執筆/第一章"
};
newPiece.Progress.Value = 0.25f;
newItem.Pieces.Add(newPiece);
```

リスト 86: jQuery

```
newPiece = $tview.timeview("addPiece", {
    itemIndex: newItem.index(),
    start: today,
    finish: new Date(today.getFullYear(), today.getMonth(), today.getDate() + 5),
    caption: "執筆/第一章"
});
newPiece.backgroundColor("pink");
newPiece.borderColor("red");
newPiece.progress().value(0.25);
```

Start プロパティ

ピースの開始日時を取得または設定します。

定義

public DateTime Start { get; set; }

プロパティ値

ピースの開始日時。Start プロパティの値は、Finish プロパティの値以下を設定します。

例

次の例は、選択したピースの開始日時を 1 日前倒しします。

リスト 87: ASP.NET C#

```
protected void PieceStartEarlyOneDayButton_Click(object sender, EventArgs e)
{
    if (KnTView1.Selection.Pieces.Count > 0)
    {
        var piece = KnTView1.Selection.Pieces[0];

        piece.Start -= TimeSpan.FromDays(1);
    }
}
```

リスト 88: jQuery

```
$("#PieceStartEarlyOneDayButton").button().click(function () {

    var selection = $tview.timeview("selection");
    if (selection.pieces().count() > 0) {

        var piece = selection.pieces().item(0);

        var start = piece.start();
        var newStart = new Date(start.getFullYear(), start.getMonth(), start.
↪getDate() - 1);

        piece.start(newStart);
    }
});
```

Text プロパティ

ピースに表示する文字列を取得または設定します。

定義

public string Text { get; set; }

プロパティ値

ピースに表示する文字列。

例

次の例は、2 行目のピースに「カタログ作成」と表示する方法を示しています。

リスト 89: ASP.NET C#

```
KnTView1.Items[1].Pieces[0].Text = "カタログ作成";
```

リスト 90: jQuery

```
$('#kntview1').timeview("items").item(1).pieces().item(0).caption().text("カタログ作成
↪");
```

ToolTip プロパティ

ピースにマウスを重ねたときに表示する文字列を取得または設定します。

定義

public string ToolTip { get; set; }

プロパティ値

ツールチップとして表示する文字列。

例

次の例は、2 行目のピースのツールチップに「ピースの表示テキストを変更しました。」と表示する方法を示しています。

リスト 91: ASP.NET C#

```
KnTView1.Items[1].Pieces[0].ToolTip = "ピースの表示テキストを変更しました。";
```

リスト 92: jQuery

```
$('#kntview1').timeview("items").item(1).pieces().item(0)
    .caption().tooltip("ピースの表示テキストを変更しました。");
```

Value プロパティ

ピースに保持させたい任意の値を取得または設定します。

定義

public object Value { get; set; }

プロパティ値

ピースに保持する任意の値。この値はピースに紐づけて保持するのみで自動で表示することはありません。

例

次の例は、CSV に作成した宿泊予約一覧のうち、ピースで表示する 1 件の予約情報を格納したオブジェクトを Value に設定しておき、ピース クリックによる選択時に Value を参照して予約の詳細を表示します。

リスト 93: ASP.NET C#

```
protected void KnTView1_AfterPieceSelect(object sender, AfterPieceSelectEventArgs e)
{
    var reserve = e.Piece.Value as Reserve;

    ReserveNameLabel.Text = reserve.Name;
    ReserveRoomLabel.Text = reserve.Room;
    ReserveStartDateLabel.Text = reserve.StartDate.ToString("yyyy 年 M 月 d 日 dddd");
    ReserveStaysLabel.Text = reserve.Stays.ToString();
}
```

リスト 94: jQuery

```
var $tview = $('#kntview1').timeview({
    columns: [
        { text: "部屋番号", width: 100 },
    ],
    afterpieceselect: function (event, piece) {
        if (piece) {
            var reserve = piece.value();

            $('#ReserveNameLabel').text(reserve.name);
            $('#ReserveRoomLabel').text(reserve.room);
            $('#ReserveStartDateLabel').text('' +
                reserve.startDate.getFullYear() + '年' +
                (reserve.startDate.getMonth() + 1) + '月' +
                reserve.startDate.getDate() + '日');
            $('#ReserveStaysLabel').text(reserve.stays);
        }
    }
});
```

6.11 PieceSelection クラス

選択中のピース *Piece* クラス のコレクションを表します。

6.11.1 概要

タイムビュー コントロールの *Selection.Pieces* プロパティから参照します。

現在はタイムビュー コントロール内での単一選択のみ対応しています。

6.11.2 プロパティ一覧

プロパティ	内容
Count	選択済みのピース数を取得します。
Item[int]	指定したインデックスにあるピースを取得または設定します。

6.11.3 メソッド一覧

プロパティ	内容
Add	選択ピース コレクションにピースを追加します。
RemoveAt	指定したインデックスの位置にあるピースを選択ピース コレクションから削除します。

6.11.4 プロパティ

Count プロパティ

選択済みのピースの総数を取得します。

定義

public int Count { get; }

プロパティ値

int

例

次の例は、選択済みのピースの総数を取得してテキストボックスに表示します。

リスト 95: ASP.NET C#

```
PieceSelection_Count.Text = KnTView1.Selection.Pieces.Count.ToString();
```

リスト 96: jQuery

```
var selection = $('#tview1').timeview("selection");

$("#SelectionPiecesCount").text(selection.pieces().count());
```

Item プロパティ

指定したインデックスにあるピースを取得または設定します。

定義

public *Piece* クラス this[int index] { get; set; }

パラメーター

index int

取得または設定するピースの 0 から始まるインデックス番号。

プロパティ値

Piece クラス

例

次の例は、選択中のピースを取得して、Value をテキストボックスに表示します。

リスト 97: ASP.NET C#

```
var selectedPiece = KnTView1.Selection.Pieces[0];

Piece_Value.Text = selectedPiece.Value.ToString();
```

リスト 98: jQuery

```
var selection = $('#tview1').timeview("selection");

var selectedPiece = selection.pieces().item(0);
PieceValue.value = selectedPiece.value();
```

6.11.5 メソッド

Add(Piece)

ピース コレクションに指定のピースを末尾に追加します。現在は単一選択のみ対応しているため、現在選択されているピースの選択を解除して新しく追加したピースを選択状態にします。

定義

public void Add(*Piece* selectedPiece);

パラメーター

selectedPiece *Piece クラス*
追加する *Piece クラス*。

例

次の例は、2 行目のピースを選択します。

リスト 99: ASP.NET C#

```
KnTView1.Selection.Pieces.Add(KnTView1.Items[1].Pieces[0]);
```

リスト 100: jQuery

```
var selection = $('#tview1').timeview("selection");

selection.pieces().add({
    itemIndex: 1,
    pieceIndex: 0
});
```

RemoveAt(int)

指定したインデックスの位置にあるピースをピース コレクションから削除します。

定義

public void RemoveAt(int index);

パラメーター

index int
ピースを削除する位置の、0 から始まるインデックス。

例

次の例は、ピースの選択を解除します。

リスト 101: ASP.NET C#

```
KnTView1.Selection.Pieces.RemoveAt(0);
```

リスト 102: jQuery

```
var selection = $('#tview1').timeview("selection");
selection.pieces().remove(0);
```

6.12 Progress クラス

ピースを使って作成した仕事や予定の達成率を表します。

6.12.1 概要

ピース バーの百分率で指定した割合を塗りつぶします。

6.12.2 プロパティ一覧

プロパティ	内容
BackColor	達成率を塗りつぶす背景色を取得または設定します。
Value	達成率を百分率で取得または設定します。

6.12.3 プロパティ

BackColor プロパティ

達成率を塗りつぶす背景色を取得または設定します。

定義

public Color BackColor { get; set; }

プロパティ値

達成率の背景色を表す Color。

例

次の例は、達成率を黄色で表示します。

リスト 103: ASP.NET C#

```
var newItem = new Item();
KnTView1.Items.Add(newItem);

var newPiece = new Piece()
```

(次のページに続く)

(前のページからの続き)

```
{
    Start = DateTime.Today,
    Finish = DateTime.Today.AddDays(5),
    Text = "第一章/井上"
};
newPiece.Progress.Value = 0.25f;
newPiece.Progress.BackColor = Color.Yellow;
newItem.Piece.Add(newPiece);
```

リスト 104: jQuery

```
var now = new Date();
var today = new Date(now.getFullYear(), now.getMonth(), now.getDate());

var newItem = $tview.timeview("addItem");
newPiece = $tview.timeview("addPiece", {
    itemIndex: newItem.index(),
    start: today,
    finish: new Date(today.getFullYear(), today.getMonth(), today.getDate() + 5),
    caption: "第一章/井上"
});
newPiece.backgroundColor("pink");
newPiece.progress().value(0.25);
newPiece.progress().backgroundColor("yellow");
```

Value プロパティ

達成率を百分率で取得または設定します。

定義

public object Value { get; set; }

プロパティ値

達成率を表す 0 から 1 までの浮動小数点数。

例

次の例は、達成率を 25%で表示します。

リスト 105: ASP.NET C#

```
var newItem = new Item();
KnTView1.Items.Add(newItem);

var newPiece = new Piece()
{
```

(次のページに続く)

(前のページからの続き)

```
    Start = DateTime.Today,
    Finish = DateTime.Today.AddDays(5),
    Text = "第一章/井上"
};
newPiece.Progress.Value = 0.25f;
newPiece.Progress.BackColor = Color.Yellow;
newItem.Piece.Add(newPiece);
```

リスト 106: jQuery

```
var now = new Date();
var today = new Date(now.getFullYear(), now.getMonth(), now.getDate());

var newItem = $tview.timeview("addItem");
newPiece = $tview.timeview("addPiece", {
    itemIndex: newItem.index(),
    start: today,
    finish: new Date(today.getFullYear(), today.getMonth(), today.getDate() + 5),
    caption: "第一章/井上"
});
newPiece.backgroundColor("pink");
newPiece.progress().value(0.25);
newPiece.progress().backgroundColor("yellow");
```

6.13 Selection クラス

タイムビューの各要素の選択状態を表します。

6.13.1 概要

タイムビューで選択可能な要素は現在ピースのみです。

6.13.2 プロパティ一覧

プロパティ	内容
Pieces	ピースの選択コレクションを取得します。

6.13.3 プロパティ

Pieces プロパティ

ピースの選択コレクションを取得します。

定義

public *PieceSelection* Pieces { get; }

プロパティ値

PieceSelection クラス への参照です。

6.14 SpecialTimes クラス

休祝日や作業のマイルストーンの表現に利用できる特別時間帯 *SpecialTime* クラス のコレクションを表します。

6.14.1 概要

タイムビュー コントロールの *SpecialTimes* プロパティから参照します。

6.14.2 プロパティ一覧

プロパティ	内容
Count	作成済みの特別時間帯数を取得します。
Item[int]	指定したインデックスにある特別時間帯を取得または設定します。

6.14.3 メソッド一覧

プロパティ	内容
Add	特別時間帯コレクションに特別時間帯を追加します。
RemoveAt	指定したインデックスの位置にある特別時間帯を特別時間帯コレクションから削除します。

6.14.4 プロパティ

Count プロパティ

作成済みの特別時間帯の総数を取得します。

定義

public int Count { get; }

プロパティ値

int

例

次の例は、作成済みの特別時間帯の総数を取得してテキストボックスに表示します。

リスト 107: ASP.NET C#

```
SpecialTimes_Count.Text = KnTView1.SpecialTimes.Count.ToString();
```

リスト 108: jQuery

```
var specialTimes = $('#tview1').timeview("specialtimes");

$("#SpecialTimesCount").text(specialTimes.count());
```

Item プロパティ

指定したインデックスにある特別時間帯を取得または設定します。

定義

public *SpecialTime* クラス this[int index] { get; set; }

パラメーター

index int

取得または設定する特別時間帯の 0 から始まるインデックス番号。

プロパティ値

SpecialTime クラス

例

次の例は、インデックスが 1 番の特別時間帯を取得して、Text をテキストボックスに表示します。

リスト 109: ASP.NET C#

```
var specialTime = KnTView1.SpecialTimes[1];

SpecialTime_Text.Text = specialTime.Text;
```

リスト 110: jQuery

```
var specialTime = $('#tview1').timeview("specialTimes").item(1);

SpecialTimeText.value = specialTime.text();
```

6.14.5 メソッド

Add(SpecialTime)

特別時間帯コレクションに指定の特別時間帯を末尾に追加します。

定義

public void Add(*SpecialTime* specialTime);

パラメーター

specialTime ***SpecialTime* クラス**
追加する *SpecialTime* クラス。

例

次の例は、5 月 5 日を黄緑色で「こどもの日」というタイトルの特別時間帯を追加します。

リスト 111: ASP.NET C#

```
using Knowlbo.Component.Web.TimeView;
using System;

namespace KnTViewSample1
{
    public partial class SpecialTimesText : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                KnTView1.SpecialTimes.Add(new SpecialTime(
                    new DateTime(DateTime.Today.Year, 5, 5),
                    "こどもの日",
                    Color.PaleGreen));
            }
        }
    }
}
```

リスト 112: jQuery

```
var specialTimes = $('#tview1').timeview("specialTimes");

specialTimes.add('date', '2023/05/05', ' こどもの日', 'palegreen');
```

RemoveAt(int)

指定したインデックスの位置にある特別時間帯を特別時間帯コレクションから削除します。

定義

public void RemoveAt(int index);

パラメーター

index int

特別時間帯を削除する位置の、0 から始まるインデックス。

例

次の例は、インデックスが 1 番の位置にある特別時間帯を削除します。

リスト 113: ASP.NET C#

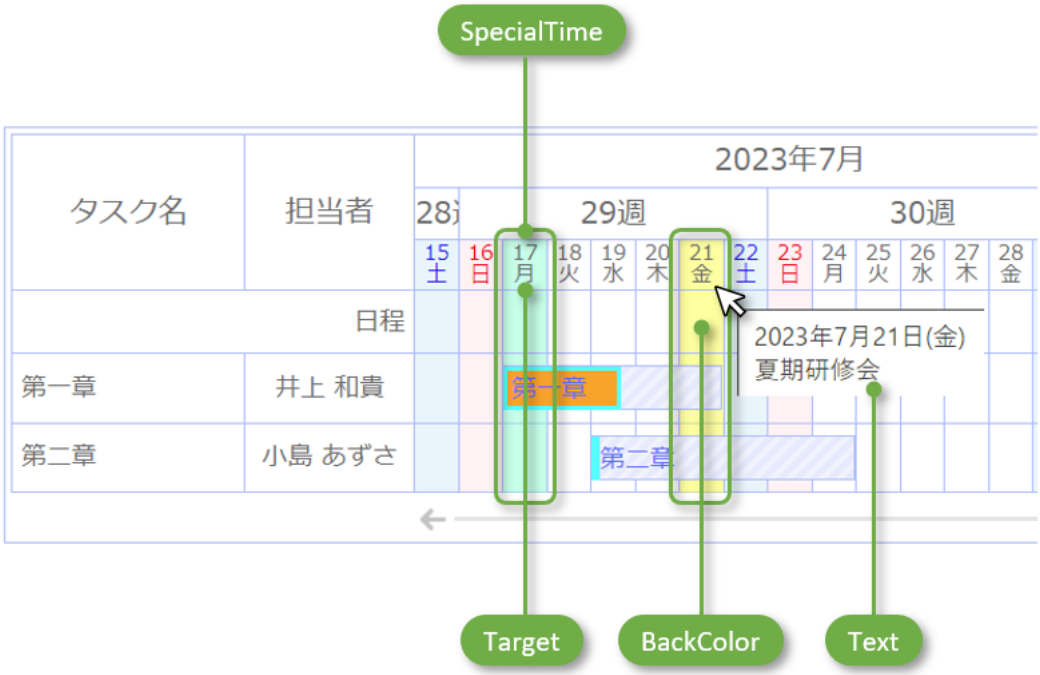
```
KnTView1.SpecialTimes.RemoveAt(1);
```

リスト 114: jQuery

```
$('#tview1').timeview("specialTimes").remove(1);
```

6.15 SpecialTime クラス

休祝日や作業のマイルストーンの表現に利用できる特別時間帯を表します。



6.15.1 概要

SpecialTime クラスは、特別時間帯を表します。Target に指定した日付全体（タイムルーラーから各アイテムを縦断した列全体に相当する領域）を BackColor に指定した色で塗り分けます。タイムルーラー上で特別時間帯を設定した日付にマウスを移動すると Text に設定した文字列をツールチップで表示します。

6.15.2 コンストラクター一覧

コンストラクタ	内容
SpecialTime()	SpecialTime クラスの新しいインスタンスを初期化します。
SpecialTime(Object, String, Color)	SpecialTime クラスの新しいインスタンスを指定のプロパティ値で初期化します。

6.15.3 プロパティ一覧

プロパティ	内容
Target	特別時間帯を適用する日付を取得または設定します。
BackColor	特別時間帯の領域を塗りつぶす色を取得または設定します。
Text	特別時間帯にマウスを重ねたときに表示する文字列を取得または設定します。

6.15.4 コンストラクター

SpecialTime クラスの新しいインスタンスを初期化します。

SpecialTime()

SpecialTime クラスの新しいインスタンスを初期値で作成します。有効に用いるには、Target、BackColor、Text の各プロパティの設定が別途必要です。

定義

public void SpecialTime();

例

次の例は、既定のコンストラクタにて特別時間帯を作成するとともに、各プロパティへ初期値を設定した特別時間帯を追加します。

リスト 115: ASP.NET C#

```
protected void Page_Load(Object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.SpecialTimes.Add(new SpecialTime() {
            Target = new DateTime(2023, 7, 21),
            Text = "夏期研修会",
            BackColor = Color.FromArgb(128, Color.Yellow)
        });
    }
}
```

SpecialTime(Object, String, Color)

SpecialTime クラスの新しいインスタンスを対象日付、塗りつぶし色、表示テキストを明示的に設定して初期化します。

パラメーター

Target Object
特別時間帯を設定する日付。

BackColor Color
特別時間帯の領域を塗りつぶす色。

Text String
特別時間帯にマウスを重ねたときに表示する文字列。

例

次の例は、社内行事「夏期研修会」が行われる 7 月 21 日 を薄黄色で塗りつぶす特別時間帯を追加します。

リスト 116: ASP.NET C#

```
protected void Page_Load(Object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.SpecialTimes.Add(new SpecialTime(
            new DateTime(2023, 7, 21),
            "夏期研修会",
            Color.FromArgb(128, Color.Yellow)
        ));
    }
}
```

リスト 117: jQuery

```
var specialTimes = $('#kntview1').timeview("specialTimes");

var specialTime = specialTimes.add('date',
    '2023/07/21', '夏期研修会', 'rgba(255, 255, 0, 0.5)');
```

6.15.5 プロパティ

Target プロパティ

特別時間帯を適用する日付を設定します。値の取得も可能です。

定義

public Object Target { get; set; }

プロパティ値

特別時間帯を適用する日付。Object 型ですが、実際には DateTime 型の値を設定願います。

例

次の例は、2023/5/5「こどもの日」を特別時間帯として表示します。

リスト 118: ASP.NET C#

```
var specialTime = new SpecialTime();
specialTime.Target = new DateTime(2023, 5, 5);
specialTime.Text = "こどもの日";
specialTime.BackColor = Color.LightYellow;
KnTView1.SpecialTimes.Add(specialTime);
```

BackColor プロパティ

特別時間帯の領域を塗りつぶす色を設定します。値の取得も可能です。

定義

public Color BackColor { get; set; }

プロパティ値

特別時間帯の塗りつぶす色を表す Color。

例

次の例は、2023/5/5「こどもの日」を薄い空色で塗りつぶす特別時間帯を設定します。

リスト 119: ASP.NET C#

```
var specialTime = new SpecialTime();
specialTime.Target = new DateTime(2023, 5, 5);
specialTime.Text = "こどもの日";
specialTime.BackColor = Color.LightSkyBlue;
KnTView1.SpecialTimes.Add(specialTime);
```

Text プロパティ

特別時間帯にマウスを重ねたときに表示する文字列を設定します。値の取得も可能です。

定義

public string Text { get; set; }

プロパティ値

特別時間帯にツールチップで表示する文字列。

例

次の例は、1 つ目の特別時間帯の各種プロパティ値を表示する方法を示しています。

リスト 120: ASP.NET C#

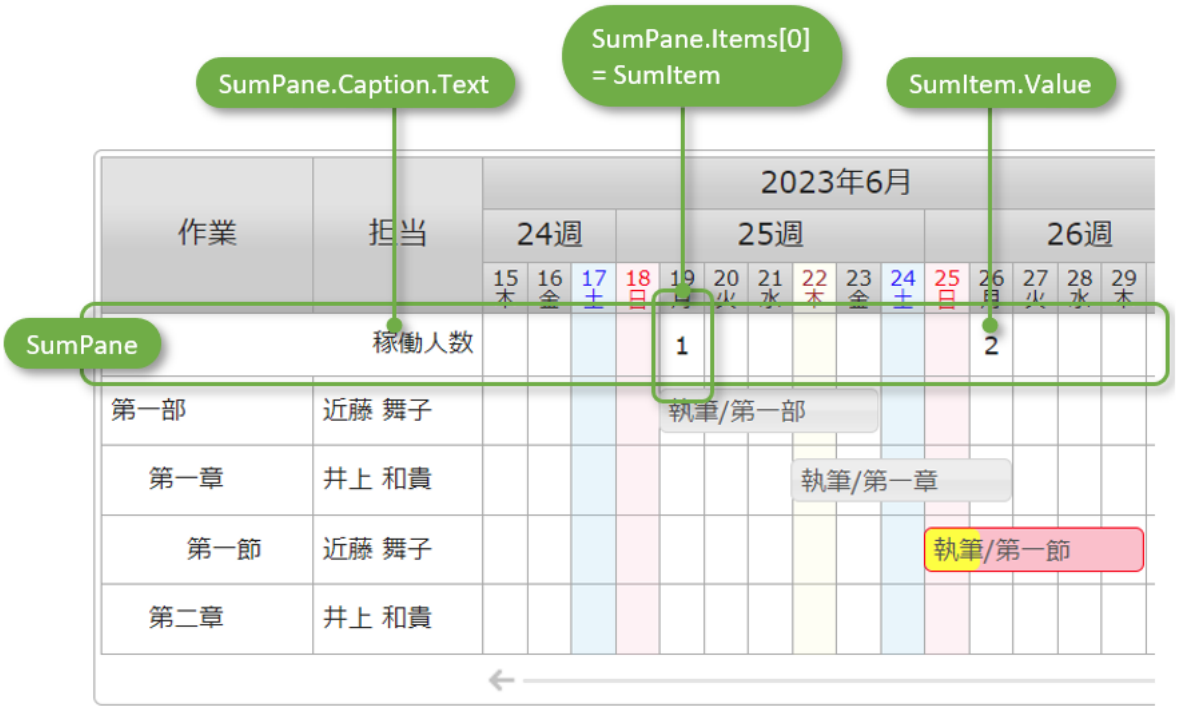
```
protected void GetSpecialTimeButton_Click(object sender, EventArgs e)
{
    var specialTime = KnTView1.SpecialTimes[0];

    SpecialTime_Target.Text =
        ((DateTime)specialTime.Target).ToString("yyyy-MM-dd");
    SpecialTime_Text.Text = specialTime.Text;
    SpecialTime_BackColor.Text =
        ColorTranslator.ToHtml(specialTime.BackColor);
}
```

リスト 121: jQuery

```
$("#GetSpecialTimeButton").button().click(function () {
    var specialTimes = $('#tview1').timeview("specialTimes");
    var specialTime = specialTimes.item(0);

    SpecialTimeDate.value =
        tview.formatDateTime("yyyy/m/d", specialTime.target());
    SpecialTimeText.value = specialTime.text();
    SpecialTimeBackground.value =
        specialTime.backgroundColor();
});
```

6.16 SumPane クラス

集計表示ペイン。アイテム領域の上部に集計値などを表示する領域を表します。



6.16.1 概要

集計表示ペインには、集計アイテムとして日付ごとの集計値やマイルストーンを表示することができます。

6.16.2 プロパティ一覧

プロパティ	内容
Caption	集計表示ペインのタイトル領域を表すオブジェクトを取得します。
Height	集計表示ペインの高さを取得または設定します。
Hidden	集計表示ペインを非表示にするかどうかを取得または設定します。
Items	集計表示ペインに表示する集計アイテムのコレクションを取得します。

6.16.3 プロパティ

Caption プロパティ

集計表示ペインのタイトル領域を表すオブジェクトを取得します。

定義

public *SumPaneCaption* Caption { get; }

プロパティ値

SumPaneCaption クラス への参照です。

例

次の例は、集計表示ペインのタイトルを「稼働人数」に設定します。

リスト 122: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.SumPane.Hidden = false;
        KnTView1.SumPane.Caption.Text = "稼働人数";
    }
}
```

リスト 123: jQuery

```
const $tview = $('#kntview1').timeview({
    columns: [
        { text: "作業", width: 100 },
        { text: "担当", width: 80 }
    ]
});

const sumPane = $tview.timeview("sumPane");

sumPane.hidden(false);
sumPane.caption().text("稼働人数");
```

Height プロパティ

集計表示ペインの高さを取得または設定します。

定義

public int Height { get; set; }

プロパティ値

集計表示ペインの高さをピクセル数で示します。

例

集計表示ペインの高さを 50 ピクセルにします。

リスト 124: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.SumPane.Hidden = false;
        KnTView1.SumPane.Height = 50;
    }
}
```

リスト 125: jQuery

```
const $tview = $('#kntview1').timeview({
    columns: [
        { text: "作業", width: 100 },
        { text: "担当", width: 80 }
    ]
});
```

(次のページに続く)

(前のページからの続き)

```
const sumPane = $tview.timeview("sumPane");

sumPane.hidden(false);
sumPane.height(50);
```

Hidden プロパティ

集計表示ペインを非表示にするかどうかを取得または設定します。

定義

public bool Hidden { get; set; }

プロパティ値

集計表示ペインを非表示にする場合は true 、表示する場合は false を設定します。既定値は true です。

例

次の例は、集計表示ペインを表示するように設定します。

リスト 126: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.SumPane.Hidden = false;
    }
}
```

リスト 127: jQuery

```
const $tview = $('#kntview1').timeview({
    columns: [
        { text: "作業", width: 100 },
        { text: "担当", width: 80 }
    ]
});

const sumPane = $tview.timeview("sumPane");
sumPane.hidden(false);
```

Items プロパティ

集計表示ペインに表示する集計アイテムのコレクションを取得します。

定義

public *Items* SumItems { get; }

プロパティ値

SumItems クラス への参照です。

6.17 SumPaneCaption クラス

集計表示ペインのタイトル領域を表します。

6.17.1 概要

集計表示ペインのうち列見出し領域にあたる場所に集計名などの文字列を表示できます。

6.17.2 プロパティ一覧

プロパティ	内容
Text	集計表示ペインのタイトル領域に表示する文字列を取得または設定します。

6.17.3 プロパティ

Text プロパティ

集計表示ペインのタイトル領域に表示する文字列を取得または設定します。

定義

public String Text { get; set; }

プロパティ値

集計表示ペインのタイトル領域に表示する任意の文字列。

例

次の例は、集計表示ペインのタイトル領域へ「稼働人数」という文字列を書きように途中で改行して表示するように設定します。

稼働
人数

リスト 128: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.SumPane.Hidden = false;
        KnTView1.SumPane.Caption.Text = "稼働\n 人数";
    }
}
```

リスト 129: jQuery

```
const $tview = $('#kntview1').timeview({
    columns: [
        { text: "作業", width: 100 },
        { text: "担当", width: 80 }
    ]
});

const sumPane = $tview.timeview("sumPane");

sumPane.hidden(false);
sumPane.caption().text("稼働\n 人数");
```

6.18 SumItems クラス

日付ごとの集計値やマイルストーンの表示に利用できる集計アイテム *SumItem* クラス のコレクションを表します。

6.18.1 概要

集計表示ペインの *SumItems* プロパティから参照します。

6.18.2 プロパティ一覧

プロパティ	内容
Count	作成済みの集計アイテム数を取得します。
Item[int]	指定したインデックスにある集計アイテムを取得または設定します。

6.18.3 メソッド一覧

プロパティ	内容
Add	集計アイテム コレクションに集計アイテムを追加します。
RemoveAt	指定したインデックスの位置にある集計アイテムを集計アイテム コレクションから削除します。

6.18.4 プロパティ

Count プロパティ

作成済みの集計アイテムの総数を取得します。

定義

public int Count { get; }

プロパティ値

int

例

次の例は、作成済みの集計アイテムの総数を取得してテキストボックスに表示します。

リスト 130: ASP.NET C#

```
SumItems_Count.Text = KnTView1.SumPane.Items.Count.ToString();
```

リスト 131: jQuery

```
var sumPane = $('#tview1').timeview("sumPane");

$("#SumItemsCount").text(sumPane.items().count());
```

Item プロパティ

指定したインデックスにある集計アイテムを取得または設定します。

定義

public *SumItem* クラス this[int index] { get; set; }

パラメーター

index int

取得または設定する集計アイテムの 0 から始まるインデックス番号。

プロパティ値

SumItem クラス

例

次の例は、インデックスが 1 番の集計アイテムを取得して、Value をテキストボックスに表示します。

リスト 132: ASP.NET C#

```
var sumItem = KnTView1.SumPane.Items[1];

SumItem_Value.Text = sumItem.Value.ToString();
```

リスト 133: jQuery

```
var sumPane = $('#tview1').timeview("sumPane");

var sumItem = sumPane.items().item(1);
SumItemValue.value = sumItem.value();
```

6.18.5 メソッド

Add(SumItem)

集計アイテム コレクションに指定の集計アイテムを末尾に追加します。

定義

public void Add(*SumItem* sumItem);

パラメーター

sumItem *SumItem* クラス
追加する *SumItem* クラス。

例

次の例は、5 月 31 日の稼働人数として 2 を表示する集計アイテムを追加します。

リスト 134: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.SumPane.Hidden = false;
        KnTView1.SumPane.Caption.Text = "稼働人数";

        KnTView1.SumPane.Items.Add(new SumItem(
            new DateTime(DateTime.Today.Year, 5, 31),
            2));
    }
}
```

リスト 135: jQuery

```
const $tview = $('#kntview1').timeview({
    columns: [
        { text: "作業", width: 100 },
        { text: "担当", width: 80 }
    ]
```

(次のページに続く)

(前のページからの続き)

```
});

const sumPane = $tview.timeview("sumPane");
sumPane.hidden(false);
sumPane.caption().text("稼働人数");

sumPane.items().add('2023/05/31', 2);
```

RemoveAt(int)

指定したインデックスの位置にある集計アイテムを集計アイテム コレクションから削除します。

定義

public void RemoveAt(int index);

パラメーター

index int

集計アイテムを削除する位置の、0 から始まるインデックス。

例

次の例は、インデックスが 1 番の位置にある集計アイテムを削除します。

リスト 136: ASP.NET C#

```
KnTView1.SumPane.Items.RemoveAt(1);
```

リスト 137: jQuery

```
var sumPane = $('#tview1').timeview("sumPane");
sumPane.items().remove(1);
```

6.19 SumItem クラス

日付ごとの集計値やマイルストーンの表示に利用できる集計アイテムを表します。

6.19.1 概要

SumItem クラスは、集計アイテムを表します。Time に指定した日付に Value で指定した任意の値を表示します。

6.19.2 コンストラクター一覧

コンストラクタ	内容
SumItem()	SumItem クラスの新しいインスタンスを初期化します。
SumItem(DateTime, Object)	SumItem クラスの新しいインスタンスを指定のプロパティ値で初期化します。

6.19.3 プロパティー一覧

プロパティ	内容
Time	集計アイテムを表示する日付を取得または設定します。
Value	集計アイテムに表示する任意の値を取得または設定します。

6.19.4 コンストラクター

SumItem クラスの新しいインスタンスを初期化します。

SumItem()

SumItem クラスの新しいインスタンスを初期値で作成します。有効に用いるには、作成後に Time、Value の各プロパティの設定が必要です。

定義

public void SumItem();

例

次の例は、既定のコンストラクタにて集計アイテムを作成するとともに、各プロパティへ初期値を設定した集計アイテムを追加します。

リスト 138: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.SumPane.Hidden = false;
        KnTView1.SumPane.Caption.Text = "稼働人数";

        KnTView1.SumPane.Items.Add(new SumItem()
        {
            Time = new DateTime(DateTime.Today.Year, 5, 31),
            Value = 2
        });
    }
}
```

SumItem(DateTime, Object)

SumItem クラスの新しいインスタンスを表示先の日付、および表示する値を明示的に設定して初期化します。

パラメーター

Time DateTime
集計アイテムを表示する日付。

Value Object
集計アイテムに表示する任意の値。

例

次の例は、5 月 31 日の稼働人数として 2 を表示する集計アイテムを追加します。

リスト 139: ASP.NET C#

```
KnTView1.SumPane.Items.Add(new SumItem(
    new DateTime(DateTime.Today.Year, 5, 31),
    2));
```

リスト 140: jQuery

```
const sumPane = $tview.timeview("sumPane");

sumPane.items().add('2023/05/31', 2);
```

6.19.5 プロパティ

Time プロパティ

集計アイテムを表示する日付を設定します。値の取得も可能です。

定義

public DateTime Time { get; set; }

プロパティ値

表敬表ペイン上の指定した日付の位置にこの集計アイテムを表示します。

Value プロパティ

集計アイテムに表示する任意の値を設定します。値の取得も可能です。

定義

public Object Value { get; set; }

プロパティ値

集計アイテムに表示する任意の値。

例

次の例は、1 つ目の集計アイテムの各種プロパティ値を表示する方法を示しています。

リスト 141: ASP.NET C#

```
protected void GetSumItemButton_Click(object sender, EventArgs e)
{
    int index;
    int.TryParse(SumItem_Index.Text, out index);
    if (!(index < KnTView1.SumPane.Items.Count))
    {
        return;
    }
```

(次のページに続く)

(前のページからの続き)

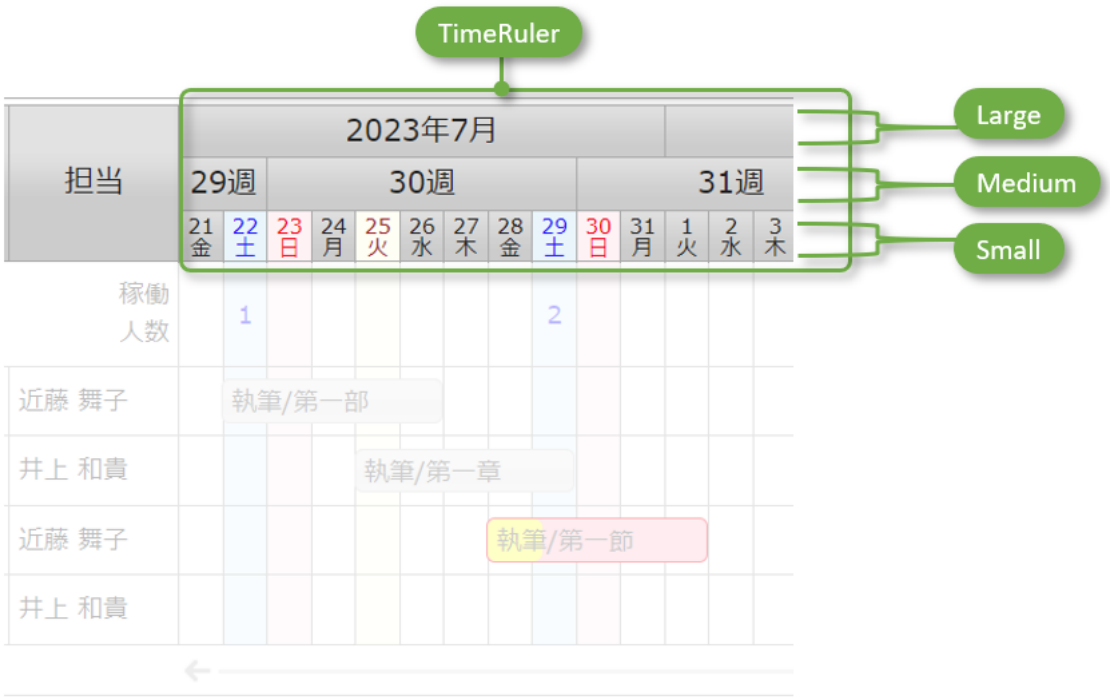
```
    var sumItem = KnTView1.SumPane.Items[index];

    SumItem_Time.Text = ((DateTime)sumItem.Time).ToString("yyyy-MM-dd");
    SumItem_Value.Text = sumItem.Value.ToString();
}
```

リスト 142: jQuery

```
$("#GetSumItemButton").button().click(function () {
    var sumPane = $('#tview1').timeview("sumPane");

    var sumItem = sumPane.items().item(SumItemIndex.value);
    if (sumItem) {

        SumItemTime.value = $('#tview1').timeview("formatDateTime",
            "yyyy/m/d", sumItem.time());
        SumItemValue.value = sumItem.value() ? sumItem.value() : "";
    }
});
```

6.20 TimeRuler クラス

タイムルーラーは、タイム ビュー コントロールの右上に表示する、時間軸の定規を表します。



6.20.1 概要

タイム ルーラーの目盛りは、大区分、中区分、小区分の 3 段で構成されます。各区分の目盛りの単位は、年、月、週、日のいずれかを指定できます。また必要に応じて、大区分または中区分を非表示にできます。

6.20.2 プロパティ一覧

プロパティ	内容
FinishTime	タイム ルーラー全体の末尾の日時を取得または設定します。
Large	上段の大区分目盛りを取得します。
Medium	中段の中区分目盛りを取得します。
Small	下段の小区分目盛りを取得します。
StartTime	タイム ルーラー全体の先頭の日時を取得または設定します。
VisibleStartTime	タイム ルーラーの画面に見える左端の日時を取得または設定します。

6.20.3 プロパティ

FinishTime プロパティ

タイム ルーラー全体の末尾の日時を取得または設定します。

定義

public DateTime FinishTime { get; set; }

プロパティ値

タイム ルーラー全体の先頭の日時。FinishTime プロパティには、SartTime プロパティよりあとの日時を設定してください。

例

全体の期間を 2023 年 4 月 1 日 0 時 0 分から 2024 年 4 月 1 日 0 時 0 分にします。日付としてみた場合に末尾の 2024 年 3 月 31 日をしっかり表示するには、FinishTime に 2024 年 4 月 1 日を指定する必要があります。

リスト 143: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Ruler.StartTime = new DateTime(2023, 4, 1);
        KnTView1.Ruler.FinishTime = new DateTime(2024, 4, 1);
    }
}
```

リスト 144: jQuery

```
const $tview = $('#kntview1').timeview({
    timeRuler: {
        start: '2023/4/1',
        finish: '2024/4/1',
    }
});
```

Large プロパティ

上段の大区分目盛りを表すオブジェクトを取得します。

定義

public *LetterRuler* Large { get; }

プロパティ値

LetterRuler クラス への参照です。

例

次の例は、大区分を非表示にします。

リスト 145: ASP.NET C#

```
protected void SetRulerScaleButton_Click(object sender, EventArgs e)
{
    KnTView1.Ruler.Large.Hidden = true;
}
```

リスト 146: jQuery

```
$('#SetRulerOption').button().click(function() {

    $tview.timeview("option", "timeRuler", {
        large: {
            hidden: true
        }
    });
}
```

Medium プロパティ

中段の中区分目盛りを表すオブジェクトを取得します。

定義

public *LetterRuler* Medium { get; }

プロパティ値

LetterRuler クラス への参照です。

Small プロパティ

下段の小区分目盛りを表すオブジェクトを取得します。

定義

public *LetterRuler* Small { get; }

プロパティ値

LetterRuler クラス への参照です。

StartTime プロパティ

タイム ルーラー全体の先頭の日時を取得または設定します。

定義

public DateTime StartTime { get; set; }

プロパティ値

タイム ルーラー全体の先頭の日時。StartTime プロパティには、FinishTime プロパティより前の日時を設定してください。

例

FinishTime の例を参照。

VisibleStartTime プロパティ

タイム ルーラーの画面に見える左端の日時を取得または設定します。

定義

public DateTime VisibleStartTime { get; set; }

プロパティ値

VisibleStartTime プロパティに設定した値は、小区分の目盛りのうち、一番近い目盛りの日時に補正されます。

例

タイム ルーラーの現在の表示の左端を 2023 年 5 月 10 日にします。

リスト 147: ASP.NET C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        KnTView1.Ruler.StartTime = new DateTime(2023, 4, 1);
        KnTView1.Ruler.FinishTime = new DateTime(2024, 4, 1);
        KnTView1.Ruler.VisibleStartTime = new DateTime(2023, 5, 10);
    }
}
```

リスト 148: jQuery

```
const $tview = $('#kntview1').timeview({
    timeRuler: {
        start: '2023/4/1',
        finish: '2024/4/1',
        visibleStartTime: '2023/5/10',
    }
});
```

6.21 PieceAddMode 列挙型

各アイテムに対しマウス操作によってピースをいくつ追加できるかを示します。

6.21.1 列挙値

名前	値	内容
Disabled	0	追加できない
Single	1	1 つだけ追加できる
Multiple	2	複数追加できる

6.22 ScaleUnit 列挙型

タイム ルーラーの各目盛りの単位を示します。

6.22.1 列挙値

名前	値	内容
Year	0	年単位
Month	1	月単位
Week	2	週単位
Day	3	日単位

6.23 SelectionMode 列挙型

ピースの選択動作を定義します。

6.23.1 列挙値

名前	値	内容
Single	1	全体で 1 つだけ選択できる

6.24 SizeMode 列挙型

タイムビュー コントロールの表示サイズの決め方を表します。

6.24.1 列挙値

名前	値	内容
Manual	0	手動サイズ。コントロールの Width、および Height で指定したサイズ。収まらない内容はスクロールで表示されます。
Auto	1	自動サイズ。内容をすべてスクロールなしで表示できるように幅、および高さを自動で広げます。
AutoWidth	2	幅のみ自動サイズ
AutoHeight	3	高さのみ自動サイズ

6.25 TvTextAlign 列挙型

文字列の水平方向の配置を示します。

6.25.1 列挙値

名前	値	内容
Left	0	左寄せ
Center	1	中央寄せ
Right	2	右寄せ