

```

%!
% NeWS Forth HyperTIES
% Window class definitions
% Don Hopkins

systemdict begin

/TIESWindow DefaultWindow
dictbegin
  /NewCan null def
  /PageLabel (No Pages) def
  /Pages [] def
  /PageNumber -1 def
  /PageDict null def
  /FreePages null def
dictend
classbegin
  /FrameFont /Times-Italic findfont 18 scalefont def
  /FrameLabel (NeWS Forth TIES) def
  /IconImage /no_ties def
  /ClientMinHeight 32 def
  /ClientMinWidth 32 def
  /BorderTop 30 def

  /current-page { % - => can
    Pages length 0 eq {null} {
      Pages PageNumber get
    } ifelse
  } def

  /next-page {
    PageNumber 1 add goto-page
  } def

  /back-page {
    PageNumber 1 sub goto-page
  } def

  /first-page {
    0 goto-page
  } def

  /last-page {
    Pages length 1 sub goto-page
  } def

  /goto-page { % page => -
    0 max Pages length 1 sub min
    /PageNumber exch def
    PageNumber -1 ne {
      current-page canvastotop
      /PageLabel (Page % of %) [PageNumber 1 add Pages length] sprintf def
      current-page dup /Mapped true put
      % unmap all pages underneath the current page
      { /CanvasBelow get dup null eq {pop exit} if
        dup /Mapped false put } loop
    } {
      /PageLabel (No Pages) def
    }
  }

```

```

    } ifelse
    paintframelabel
    WinPileID ContentsPileID eq {
        update-page-trackers } if
} def

/update-page-trackers {
    % Tell every target tracking PageNumber that it's changed.
    createevent begin
        /Name /Page def
        /Action PileDict WinPileID get /ContentsPileID get def
        /ClientData 10 dict def
        ClientData begin
            /PageNumber PageNumber def
            /PageCount Pages length def
        end
        currentdict end sendevent
    } def

/new-canvas { % - => can
    FreePages length 0 eq {
        ClientCanvas newcanvas
    } {
        FreePages {exit} forall
        FreePages exch undef
    } ifelse
    /NewCan exch def
    /Pages [Pages aload pop NewCan] def
    NewCan /Transparent false put
    NewCan /Retained true put
    NewCan /EventsConsumed /MatchedEvents put
    ClientCanvas setcanvas
    clippath NewCan reshapecanvas
    NewCan canvastobottom
    gsave NewCan setcanvas ClientFillColor fillcanvas grestore
    NewCan
    /NewCan null def
    PageNumber goto-page
    5 { pause } repeat
} def

% so we don't repaint page when frame damaged
/CreateFrameCanvas {
    /CreateFrameCanvas super send
    FrameCanvas /Retained true put
} def

/wipe-page { % Can => -
    PageDict exch get begin
        currentdict /ItemMgr known {
            ItemMgr type /processtype eq {
                ItemMgr killprocess
            } if
        } if
    Targets { % TID Ref
        pause pause
        pop TargetDict 1 index known {
            % Destroy the target object

```

```

        TargetDict 1 index get % TID Obj
        dup /ItemEventManager get % TID Obj Mgr
        currentprocess ne { % Let's not shoot ourselves in the foot ...
% errordict won't be on top of dict stack in the send context!
%         mark { /destroy 2 index send } errored cleartomark
% may cause leaks due to defs into errordict on top of dict stack
%         mark { {destroy} errored pop } 2 index send cleartomark
        mark
        {/destroy where} 2 index send {
            pop /destroy 2 index send
        } if
        cleartomark
    } {
%         (Saved from death: % %\n) [TID Ref] dbgprintf
    } ifelse
    pop % TID
    % Remove the target from TargetDict
    TargetDict exch undef %
    } {pop} ifelse
} forall
pause
[Targets {pop} forall] {Targets exch undef} forall
Can /TopChild get {
    pause
    dup null eq {pop exit} if
    dup /Retained false put
    dup /Mapped false put
    % revoke all the interests?
    /CanvasBelow get
} loop
end
} def

/unmap {
    Pages length 0 ne Iconic? not and {
        ClientCanvas /TopChild get /BottomCanvas get
        { dup /Mapped false put
            /CanvasAbove get
            dup null eq { exit } if
        } loop
        pop
    } if
    /unmap super send
} def

/map {
    /map super send
    Pages length 0 ne Iconic? and {
        ClientCanvas /TopChild get
        { dup null eq { exit } if
            dup /Mapped true put
            /CanvasBelow get
        } loop
        pop
    } if
} def

/zap-pages {

```

```

Pages {
    pause pause
    dup wipe-page
    FreePages 1 index dup put
    dup /Mapped false put
    dup gsave setcanvas ClientFillColor fillcanvas grestore
    dup /Transparent true put
    PageDict exch undef
} forall
/Pages [] def
-1 goto-page
/Can null store
/Page null store
ClientCanvas setcanvas
} def

/destroy {
    unmap
    Pages length 0 ne { zap-pages } if
%   PileDict WinPileID get Pile eq {/Pile null store} if
    PileDict {
        /Win get self eq {
            % This may not be cool to do inside of "PileDict {...} forall"!
            PileDict exch undef
            % Do some sort of callback here to tell TIES that this pile
            % was destroyed. (.destroy-pile <WinPileID>\n)
            exit
        } { pop } ifelse
    } forall
    /destroy super send
} def

/new {
    /new super send begin
    /ClientMenu [
        (Return) { ContentsPileID (.pile % .return\n) sprintf callback }
        (Next) { /next-page PileDict ContentsPileID get /Win get send }
        (Index) { ContentsPileID (.pile % .index !index\n) sprintf
                    callback }
        (Back) { /back-page PileDict ContentsPileID get /Win get send }
    ] /new DefaultMenu send def
%   ClientMenu /MenuPileID ContentsPileID put
    /PageDict 100 dict def
    /FreePages 100 dict def
    currentdict end
} def

/paint-canvas { % damage? can => -
    PageDict 1 index known {
        PageDict exch get
        begin
            gsave
            Can setcanvas
            {damagepath clip} if
            newpath
            /DL load exec
            grestore
        end
    }

```

```

    } {pop pop} ifelse
  } def

/damage-paint-canvas { % can => -
  true exch paint-canvas
} def

/PaintClient {
  Pages length 0 ne {
    false current-page paint-canvas
  } if
} def

/PaintFocus {} def

/set-name {
  paintframelabel
  /FrameLabel exch def
  paintframelabel
} def

/PaintFrameLabel {
  FrameHeight BorderTop sub
  BorderTop currentfont fontascent sub 2 div
  currentfont fontdescent
  max 1 add add

  gsave
    BorderLeft FrameHeight BorderTop sub 1 add
    FrameWidth BorderRight sub 2 add BorderLeft sub BorderTop 2 sub
    rectpath
    FrameFillColor setcolor fill
  grestore

  BorderLeft 5 add 1 index moveto
  FrameLabel show

  FrameWidth BorderRight sub 5 sub exch moveto
  PageLabel rshow
} def

classend def

/TIESContentsWindow TIESWindow
dictbegin
  /BorderBottom 4 def
dictend
classbegin
classend def

/TIESDefinitionWindow TIESWindow
dictbegin
  /BorderBottom 4 def
  /FrameLabel (Definition) def
dictend
classbegin
  /PaintFrameLabel {
    FrameHeight BorderTop sub

```

```

BorderTop currentfont fontascent sub 2 div
currentfont fontdescent
max 1 add add

gsave
  BorderLeft FrameHeight BorderTop sub 1 add
  FrameWidth BorderRight sub 2 add BorderLeft sub BorderTop 2 sub
  rectpath
  FrameFillColor setcolor fill
grestore

BorderLeft 5 add exch moveto
FrameLabel show
} def
classend def

/TIESControlsWindow TIESWindow
dictbegin
  /BorderTop 4 def
  /BorderBottom 4 def
dictend
classbegin
  /PaintFrameLabel {} def
classend def

end % systemdict

```