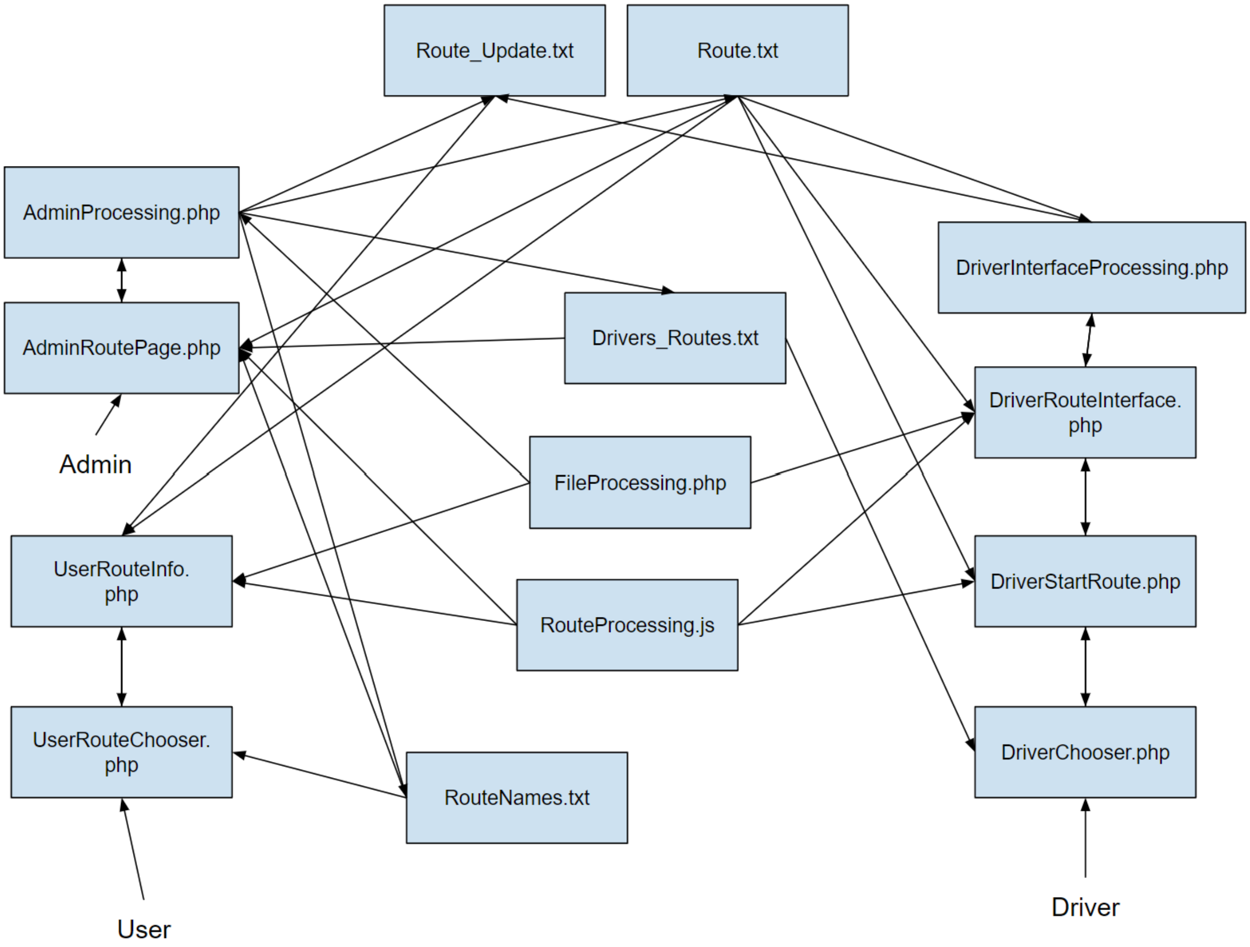
Transit Master

Documentation/Guide

By: Damian Gladysz

File Interaction Schematic



File	Purpose	Dependencies
ADMIN		
AdminRoutePage.php	Page that displays selected route by the admin along with its associated data and allows for the editing by clicking on the save button and sending the form data over to AdminProcessing.php to be written to the files	*Route*.txt (if *Route* selected) RouteNames.txt FileProcessing.php RouteProcessing.js AdminRoutePage.php
AdminProcessing.php	Is called from AdminRoutePage.php and does all the background processing for the request indicated by what the user inputted into the form on AdminRoutePage.php and then edits and writes into files to make it happen, afterwards redirecting user back to AdminRoutePage.php	AdminRoutePage.php Drivers_Routes.txt RouteNames.txt *Route*.txt (if *Route* is being edited) *Route*_Update.txt (if *Route* is being edited)
Driver		
DriverChooser.php	Allows for the selection of a driver in place of a login page and then sends user to DriverStartRoute.php	Drivers_Routes.txt DriverStartRoute.php
DriverStartRoute.php	Driver side UI displaying route and its designated schedule that was assigned to the selected driver, upon pressing start route, redirects to DriverRouteInterface.php	RouteProcessing.js FileProcessing.php DriverChooser.php DriverInterfaceProcessing.php
DriverRouteInterface.php	Displays all necessary information for driver to be able to understand where they are going next along with giving them the means to update their location at the touch of a button along with allowing to undo accidental inputs, by submitting the form and running it over to DriverInterfaceProcessing.php	DriverStartRoute.php FileProcessing.php DriverInterfaceProcessing.php RouteProcessing.js
DriverInterfaceProcessing.php	Is called from DriverRouteInterface.php or DriverStartRoute.php and does the background processing for changing the contents of the *Route*_Update.txt files along with the processing for correcting accidental inputs	DriverStartRoute.php FileProcessing.php DriverRouteInterface.php

User		
UserRouteChooser.php	Menu for user to choose between routes available	RouteNames.txt
UserRouteInfo.php	Page that displays all information pertaining to specific route selected in UserRouteChooser.php	RouteProcessing.js UserRouteChooser.php FileProcessing.php UserRouteChooser.php
Function Files		
FileProcessing.php	Stores functions that edit the files that all the pages read from	None
RouteProcessing.js	Stores the functions that have to do with dynamic object creation, time conversion and a couple of others	None
Text/Data Storage Files		
RouteNames.txt	Stores names of all routes that were saved	None
Drivers_Routes.txt	Stores all Drivers along with their assigned routes and bus number	None
Route.txt	Stores all information pertaining to the route signified in the file name and is edited by admin	None
*Route*_Update.txt	Stores status of Route indicated in the filename	None

Admin

AdminRoutePage.php

Functions Declared in File:

DataCompiler() (js)

Purpose/Inner-Workings:

Called to prepare form data whenever an admin attempts to submit a form to be sent over to AdminProcessing.php if no errors are detected, and if they are, notify the admin. Sends out an alert if no name is assigned, if no weekdays are selected for the route to run, if a stop name field has nothing inputted, if any of the time table times are out of order or if an inappropriate input was detected in the timetable. Afterwards, if no flags were tripped, the function appends 3 hidden inputs to the form before returning true, “newRouteName”, “RouteContents”, and “newBusAmt”.

CreateInputTable(rows, stops, busses, data) (js)

Purpose/Inner-Workings:

Used to create the original onload form table, all inputs are self explanatory with data either being the data of the route that was selected or 0 if a new route is being created, if data is 0, all inputs for the table will be empty, if it's not 0, then the inputs will be filled with the data of the existing route

DecStop() (js)

Purpose/Inner-Workings:

To decrease amount of columns in the form’s timetable, will not decrease below 1

IncStop() (js)

Purpose/Inner-Workings:

To increase amount of columns in the form timetable

DecRow() (js)

Purpose/Inner-Workings:

To decrease amount of rows by one for each individual bus in the form timetable

IncRow() (js)

Purpose/Inner-Workings:

To increase amount of rows by one for each individual bus in the form timetable

DecBus() (js)

Purpose/Inner-Workings:

To decrease amount of busses in the form timetable by one

IncBus() (js)

Purpose/Inner-Workings:

To increase amount of busses in the form timetable by one

delRoute() (js)

Purpose/Inner-Workings:

To create and a attach a hidden input that if set signifies the deletion of the currently looked at Route

File Process (not in order of execution, just in order that comes in the file):

1. Create form using the contents of RouteNames.txt that allows user to select between all created routes
2. If a route has been selected, set flag to 1 and read the associated routes data into a variable using a function
3. If flag is set to 1, create a hidden input in the main form(id=”routeEditor”) for route editing that contains the original name of the route in case of editing
4. Create a line of checkboxes signifying the days that the route is active (Mon, Wed, Fri, etc.)
5. If flag is set to one, use contents of file to set the correct checkboxes for the days the route is currently active
6. If flag equals one, create time table using data in pre-existing route, otherwise create an empty table
7. Create the encased table editing buttons and hook them up to their respective functions and labels

8. Create the save button and hook it up to DataCompiler()
9. Create the delete route button and hook it up to delRoute()
10. Get the amount of rows, stops and busses and put that info inside their respective divs for the table editing buttons

AdminProcessing.php

Functions Declared in File:

drfd() (php)

Purpose:

To delete all assignments of a specific route from the Drivers_Routes.txt file in case of renaming of route or its deletion

File Process (not in order of execution, just in order that comes in the file):

1. Set flag to 0
2. Check if either newRouteName or delFlag are set
    a. If either are set, program continues
    b. If not, send the process back to the AdminRoutePage.php
3. If delFlag is set, we delete the route data and update file associated with the route, remove the route name from the RouteNames file and call drfd()
4. Else if oldRouteName is set, that means we’re overwriting a route, we check if the newly sent name is the same as the old one
    a. If it is, we continue with the program
    b. If it isn’t, we delete all traces of the original route name and replace it in the RouteNames file

    We then open the files corresponding to the route and we fill them with the formatted data we received from AdminRoutePage, we then set flag to 1
5. Else we are creating a whole new route, creating new files from scratch and filling them with the content we were previously given from AdminRoutePage, after which we set flag to 1
6. We make a form that if flag was set to one, contains the name of the route we just edited or created
7. The form gets submitted and we get sent back to AdminRoutePage

Driver

DriverChooser.php

Functions Declared in File:

None

File Process (not in order of execution, just in order that comes in the file):

1. Create a form pointing towards the DriverStartRoute file
2. Using the file Drivers_Routes, in a while loop, create a column of submit buttons with the drivers names on them

DriverStartRoute.php

Functions Declared in File:

timeformswitch() (js)

Purpose:

To switch time format flag in local storage

File Process (not in order of execution, just in order that comes in the file):

1. Check if driver is set
   a. If it is, we continue the program
   b. If it isn't, we send the driver to the DriverChooser
2. Create a form that calls current file when submitted so that whenever someone presses the button, the format of all the times on the page will switch to military time or standard time by calling timeformswitch() before refreshing the page
3. Check current value of TF in local storage, If null, set TF to 1
4. Check current value of TF in local storage
   a. If TF equals 1, the time format button will say standard time
   b. Else, the time format button will say military time
5. Set flag to 0 and check if driver has route assigned using using DR()
   a. If not, no route was assigned
   b. If yes, route was assigned and change flags value to 1
6. Check if flag was set to 1, and if it is, make a form appear that allows for driver to start route
7. Print out schedule table for the individual driver
8. Check to see if page loaded with an errorNotif
   a. If yes
      i. If errorNotif equals 6, it means routes was completed
      ii. If errorNotif equals 4, it means the route wasn't scheduled for now
      iii. If errorNotif equals 1, it means the route was ended by the driver
      iv. Else it means a specific function in the code went rogue

DriverRouteInterface.php

Functions Declared in File:

DisplayOverlay() (js)

Purpose:

To display schedule overlay when button is pressed

RidOverlay() (js)

Purpose:

To get rid of schedule overlay when anything besides the back button is pressed

BackButtonCaller() (js)

Purpose:

To create a hidden element within the form and then submit it so that the bus may reverse its last input

File Process (not in order of execution, just in order that comes in the file):

1. Check if driver, route, and bus are set, and if they are not, send the driver back to DriverStartRoute.php
2. Create overlay that contains the individual bus drivers schedule and is hidden at beginning
3. If backTrackdata is set, create a button within the overlay that will allow driver to reverse last input
4. Add event listener to BackButton to make it trigger BackButtonCaller()
5. Create form which will run DriverInterfaceProcessing.php upon submission
6. Fill form with hidden inputs containing the driver, route, and bus
7. If NotifStopFlag is set, add a hidden input to the form containing the stop

DriverInterfaceProcessing.php

Functions Declared in File:

KillPage($code) (php)

Purpose:

When called is used to kill page by redirecting Driver back to DriverStartRoute via form with an error code which will get interpreted over there

File Process (not in order of execution, just in order that comes in the file):

1. Check if driver, route and bus are set, and if they are not, redirect driver to DriverStartRoute
2. Get bus route status using IBS()
3. Check if flag was set
    a. If flag was set
        i. Check if flag is 2, and if it is, end route through KillPage() code 1
        ii. Else if stop is set, and SU() returns false with stop on the end, KillPage() code 2
        iii. Check if flag is 1, and if it is, see if SU() returns, if it does, KillPage() code 3
    b. Else if backTrackdata was set
        i. Call SUO() and IBS()
            1. If bus route status equals neither nr or r, next stop = brs[1] + 1
4. Check if bus route status equals nr or r
    a. If it does equal nr, call SU() to change it to r in the file
    b. Identify first stop using FSF()
    c. If FSF() brought back false, call KillPage() code 4
5. If next stop equals false, call KillPage() code 5
6. Call SI() to identify stop
7. If StopInfo is false, end the route and call KillPage() code 6
8. Create form that leads to specified action with all the information necessary for the DriverInterface
9. Set NotifStopFlag or backTrackdata form if applicable
10. Submit form

User

UserRouteChooser.php

Functions Declared in File:

None

File Process (not in order of execution, just in order that comes in the file):

1. Create a form pointing towards the UserRouteInfo file
2. Using the file RouteNames, in a while loop, create a column of submit buttons with the routes names on them

UserRouteInfo.php

Functions Declared in File:

timeformswitch() (js)

Purpose:

To switch time format flag in local storage

File Process (not in order of execution, just in order that comes in the file):

1. Check if route is set, if not, send user to UserRouteChooser
2. If a cookie is set and the file edit time is the same as it was previously, set the flag to one
3. Get all data pertaining to route and convert it for easy use
4. Declare a back button that allows user to go back to UserRouteChooser
5. Read data and write the amount of busses active along with weekdays they're active on
6. Declare TF if not already declared and give time format switcher its label
7. Create table using CreateSchedTab() and append it to "Table"
8. Create dynamic buttons for the individual bus stops and append them
9. If flag equals 1, call FindETA() to get data on stop that was clicked replace text in designated div

Function Files

All functions require military time

FileProcessing.php
Functions Declared in File: RD($route) Purpose: Route Data Acquisition RS($route) Purpose: Route Status Acquisition IBS($route, $bus) Purpose: Individual Bus Status SR($route, $bus) Purpose: Stop Route, (puts nr for bus in status file) SI($route, $stopNum, $bus) Purpose: Stop Identifier FSF($route, $bus) Purpose: First Stop Finder, identifies first possible stop for driver SUO($route, $bus, $dataToInput) Purpose: Stop Update Override, allows for overwriting of data with anything desired SU($route, $bus, $startStop) Purpose: Stop Update, moves onto next stop in line, and calls SR() when no more stops are present DRDA() Purpose: Driver Route Data Acquisition (acquires all file contents) DR($driver) Purpose: Driver Route, identifies route driver was assigned UDA() Purpose: Unassigned Driver Acquisition, lists all drivers which have no route assigned (not tested) ADA($route, $numOfDrivers) Purpose: Assigned Driver Acquisition, gives all drivers assigned a specific route regardless of bus number AD() Purpose: All drivers and their routes in an array DD($driver) Purpose: Driver deletion, gets rid of specified driver from Drivers_Routes DA($driver) Purpose: Driver Addition, adds driver to Drivers_Routes

RouteProcessing.php

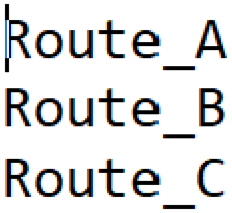
Functions Declared in File:

- CreateSchedTab(BusAmt, Rout, UniStops, Sched, Flag)
- Purpose:
  - Creates a table for the specified route including all the busses that run that route, UniStops is the amount of columns the table should have, Sched is from the route data, and Flag is a 1 or 0 dependent on what time format is wanted
- CreateSchedTabForBus(BusAmt, Bus, Rout, UniStops, Sched, Flag)
- Purpose:
  - Same as CreateSchedTab, but for individual busses
- FindETA(r, s, tstop, Flag)
- Purpose:
  - r is a dictionary created using the RouteDataDictConverter() function and s is a array created using RouteStatusConverter() with stop being the target stop, and flag being the time format to give its response in
- TimeComp(t1, t2)
- Purpose:
  - if t1 is earlier than or equal to t2, return true, else false
- TimeAdd(time, min)
- Purpose:
  - Time is the time you want to add too, and min is the amount of minutes you want to add to it
- TimeDiffMin(time1, time2)
- Purpose:
  - Left side receives later time, and it returns the amount of minutes in between those two times
- RouteDataDictConverter(data)
- Purpose:
  - Turns raw route file data into a dictionary for easier accessibility
- RouteStatusConverter(data)
- Purpose:
  - Turns raw status file data into an array
- MilitaryToStandard(time, flag)
- Purpose:
  - Turn Military to Standard time if flag is 1

Text/Data Storage Files

RouteNames.txt

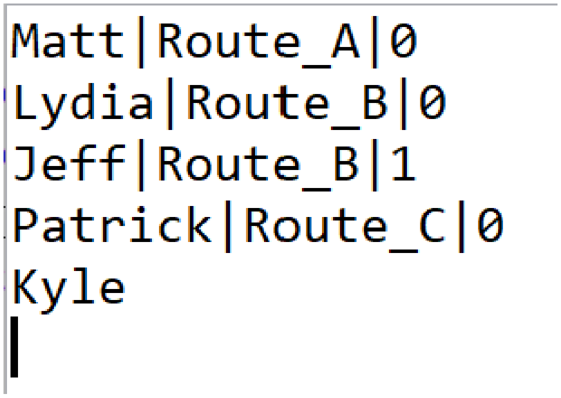
Stores all Route Names separated by newline characters with an empty line at the end



```
Route_A
Route_B
Route_C
```

Drivers_Routes.txt

Stores all driver along with their assigned route and bus number to the right of them if they have a route assigned at all, also has a new line at the end

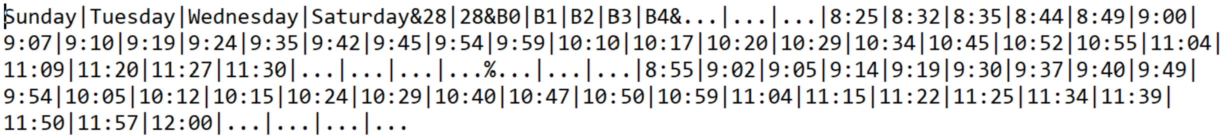


```
Matt|Route_A|0
Lydia|Route_B|0
Jeff|Route_B|1
Patrick|Route_C|0
Kyle

```

***Route*.txt**

The order of the data goes weekdays active, amount of stops on route, the names of the stops, and finally the actual schedules themselves, in the case of multiple on the same route, the weekdays and stops are shared. The amount of stops for each bus are separated by a “|” and their schedules are separated by “%”, and the weekdays and schedules are also separated by “|”



```
Sunday|Tuesday|Wednesday|Saturday&28|28&B0|B1|B2|B3|B4&...|...|...|8:25|8:32|8:35|8:44|8:49|9:00|
9:07|9:10|9:19|9:24|9:35|9:42|9:45|9:54|9:59|10:10|10:17|10:20|10:29|10:34|10:45|10:52|10:55|11:04|
11:09|11:20|11:27|11:30|...|...|...|...%...|...|...|8:55|9:02|9:05|9:14|9:19|9:30|9:37|9:40|9:49|
9:54|10:05|10:12|10:15|10:24|10:29|10:40|10:47|10:50|10:59|11:04|11:15|11:22|11:25|11:34|11:39|
11:50|11:57|12:00|...|...|...|...
```

***Route_Update*.txt**

Different busses statuses are separated by “&”, nr symbolizes that a bus is not running, r symbolizes a bus that is running but hasn’t updated their status yet,