**Basic Setup**

1. **Create a backup of your project.** I’m 96% sure nothing in my plugin will break your project, but it never hurts to back it up before installing a new plugin.
2. Install the Social Links plugin by copying it into your project’s Plugins folder.
3. Move the files in the Graphics folder included into the same respective folders in your project.
4. Explore the settings in Plugins > Social Links > 000\_Settings.
5. Define the Social Link Profiles for your game in Plugins > Social Links > 002\_Configuration.
6. To access the Social Media app, where you can view your current bond with a Social Link among other things, you need to call pbSocialMedia or pbSocialLinks. You can create a Pause Menu or Pokegear option to do this, depending on what your game setup looks like. Review the definitions of these for your game, duplicate one, and create one for Social Media using this function.
7. If you wish to customize the graphics for your game, explore Graphics > UI > Social Links.
8. If you wish to use Script Variables (to allow checking a Social Link’s bond level as a game variable for event pages) or Bond Effects (EXP and Shiny Rate boosts after you reach certain bond levels for Social Links), review the **Manual Setup – Implementing Script Variables, EXP Boosts, and Shiny Rate Boosts** section at the bottom of this document.
9. To get anything out of this plugin, you’ll need to implement Social Links throughout your game and it will be unique to each game. Review the Overview section below for a summary of how this plugin can be used.

**Overview**

The Social Links plugin is centered around bonding with NPCs in your game. As your bond increases, you can implement bonuses (or maybe disadvantages) for the player. This plugin gives you tools and it’s up to you to realize the potential. Summaries of the tools available to you are listed below. For specifics about functions (like arguments), review the Functions section later in the document.

**Social Media**

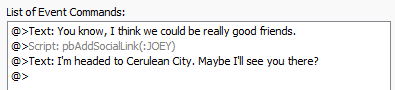
The first tool available to you is the Social Media app. By calling pbSocialMedia or pbSocialLinks, you can view all active Social Links and your current bond level, as well as social media-inspired profiles each Social Link. If you just want a basic list and don’t want to use the profile feature, you can turn it off in Settings. Features of the social media page:

* View the bond value with the Social Link.
* View the current location of the Social Link, which you can change at any time.
* View the Social Link’s favorite Pokemon, which you can change at any time.
* View the Social Link’s current status message, which you can change at any time.
* If you have the Instant Messages plugin installed, and you have conversations including the Social Link, you can use the SPECIAL key to jump to the Instant Messages app to view all conversations including them.

**Adding Social Links**

The next tool available to you is the function pbAddSocialLink, which is how you add Social Links. You’ll likely do this while talking to an NPC at a specific point in your game. In the rare case you want to remove a Social Link, you can use pbRemoveSocialLink.

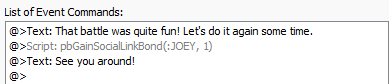
Example:



**Changing Bond Levels**

The next set of tools are functions that increase and decrease bond levels for your Social Links using pbGainSocialLinkBond, pbLowerSocialLinkBond, or pbSetSocialLinkBond. This will usually occur during important conversations with NPCs or side stories. This can range anywhere from a battle, to a major story event, to a random conversation on a park bench.

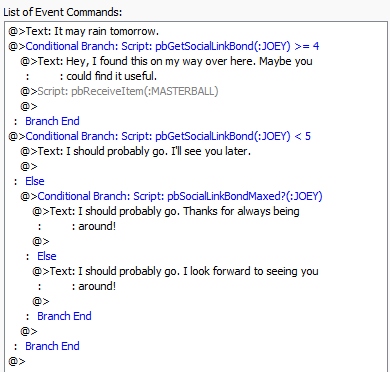
Example:

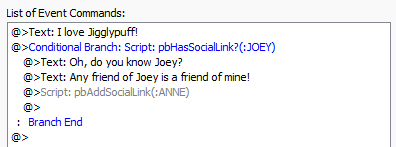


**Checking Bond Levels – Conditional Branches**

The next set of tools are used to get benefits out of bonds with Social Links. Using pbGetSocialLinkBond and pbSocialLinkBondMaxed? in Conditional Branches, you can have special events occur when chatting with a Social Link at a certain bond level. You can also use pbHasSocialLink? to have certain events occur just by having a specific active Social Link.

Examples:

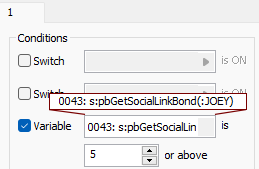


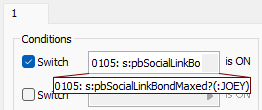


**Checking Bond Levels – Event Page Conditions**

You can also check bond levels in an event’s page conditions using Script Switches or Script Variables to make certain NPCs appear in places based on your bond with them. (To support Script Variables, you’ll need to manually add code. Refer to the **Manual Setup – Implementing Script Variables, EXP Boosts, and Shiny Rate Boosts** section towards the bottom of this document.)

Examples:





**Checking Bond Levels – Ideas**

Here are some ideas you could use checking bond levels for your game:

* After a conversation, the Social Link gives you random items based on bond level (Conditional Branch)
* Once you hit a certain bond level with a Social Link, a unique side quest becomes available (Script Variable or Conditional Branch)
* Once you hit a max bond level with a Social Link, they appear at a specific location and offers you an optional battle (Script Switch)
* Once you hit a certain bond level with a Social Link, they appear in Pokemon Centers and offer you trading opportunities (Script Variable)
* Any Social Link that you’ve maxed out will appear at the entrance of the Pokemon League to root for you (Script Switch)
* (If using the Instant Messages plugin) At certain bond levels, Social Links can send you random messages (PASSIVE\_TRIGGERS\_SEQUENCIAL\_POOL or PASSIVE\_TRIGGERS\_RANDOM\_POOL)

**Linking an Event to a Social Link**

Instead of having to remember to add the correct Social Link ID to every function you use in events, or have a ton of separate switches and variables for different Social Links, you can link an event to a specific Social Link and use “self” as the argument instead of the ID. To do so, add “SocialLink(:ID\_HERE)” to the name of the event, replacing :ID\_HERE with the Social Link ID you want to use. For example, “SocialLink(:JOEY)”. Once you do that, you can change your Script Switches/Variables to be, for example, “s:pbGetSocialLinkBond(self)” and other functions you use in the event to, for example, “pbGainSocialLinkBond(self, 1)”.

Name setup example:



**Social Media Usage**

Lastly, there are tools available to give Social Links a lively social media presence. If you don’t want to use the social media features, these won’t be useful.

* pbSetSocialLinkLocation will change the location that appears on the Social Link’s profile. This can provide the player a clue on where to find them for their next interaction, or simply a reminder.
* pbSetSocialLinkFavoritePokemon will change what Pokemon appears on the Social Link’s profile as their favorite. You could make this feel similar to Arcana from the Persona series, or simply be visual flair.
* pbSetSocialLinkStatus or pbSetSocialLinkStatusRandom will update the Social Link’s status message visible on their profile. This can be used to give more dimension to NPCs, provide hints on where you can interact with them next, etc.

**Documentation**

**Functions**

pbSocialMedia

* Description: Opens the list of active Social Links and gives access to the social media functionality. There are no arguments for this function.
* Alternate function: pbSocialLinks

pbSocialMediaDirect(SocialLinkID)

* Description: Directly opens the social media page of the specified Social Link.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to open.
* Alternate function: pbSocialLinkDirect
* Example:
  + pbSocialMediaDirect(:JOEY) => Opens the social media page for :JOEY.

pbAddSocialLink(SocialLinkID, silent: <val>)

* Description: Adds the specified Social Link. Returns **true** if the successfully added. Returns **false** if it was not added, possibly because you already have the Social Link active.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to add.
  + silent: <val>
    - Optional. Replace <val> with **true** to not show a message saying you added the Social Link. You could then add your own unique message in an event before or after you call this function.
    - If this isn’t set, it will show a message as part of the function.
* Alternate function: pbAddSL
* Examples:
  + pbAddSocialLink(:PROFOAK) => Adds :PROFOAK as a Social Link.
  + pbAddSL(:JOEY, silent: true) => Adds :JOEY as a Social Link, but does so without including a message saying so.

pbRemoveSocialLink(SocialLinkID, silent: <val>)

* Description: Removes the specified Social Link. Returns **true** if the successfully removed. Returns **false** if it was not removed, possibly because you didn’t have the Social Link yet.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to remove.
  + silent: <val>
    - Optional. Replace <val> with **true** to not show a message saying you removed the Social Link. You could then add your own unique message in an event before or after you call this function.
    - If this isn’t set, it will show a message as part of the function.
* Alternate function: pbRemoveSL
* Examples:
  + pbRemoveSocialLink(:JOEY) => Removes :JOEY from being a Social Link.
  + pbAddSL(:PROFOAK, silent: true) => Removes :PROFOAK from being a Social Link, but does so without including a message saying so.

pbHasSocialLink?(SocialLinkID)

* Description: Returns **true** if you have the Social Link active. Returns **false** if you don’t.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to check.
* Alternate function: pbHasSL?
* Examples:
  + pbHasSocialLink?(:JOEY) => Returns **true** if you have :JOEY as a Social Link.
  + pbHasSL?(:PROFOAK) => Returns **true** if you have :PROFOAK as a Social Link.

pbGainSocialLinkBond(SocialLinkID, Value, silent: <val>)

* Description: Increase the bond with specified Social Link. Returns **true** if the bond was successfully increased. Returns **false** if it was not increased.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to increase the bond with.
  + Value
    - Optional. Set to an integer representing how much you want to increase the bond by.
    - If this isn’t set, it will default to 1.
  + silent: <val>
    - Optional. Replace <val> with **true** to not show a message saying you increased the bond with the Social Link. You could then add your own unique message in an event before or after you call this function.
    - If this isn’t set, it will show a message as part of the function.
* Alternate function: pbGainSLBond
* Examples:
  + pbGainSocialLinkBond(:JOEY) => Increases the bond with :JOEY by 1.
  + pbGainSLBond(:PROFOAK, 2, silent: true) => Increases the bond with :PROFOAK by 2, but does so without including a message saying so.

pbLowerSocialLinkBond(SocialLinkID, Value, silent: <val>)

* Description: Decrease the bond with specified Social Link. Returns **true** if the bond was successfully decreased. Returns **false** if it was not decreased.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to decrease the bond with.
  + Value
    - Optional. Set to an integer representing how much you want to decrease the bond by.
    - If this isn’t set, it will default to 1.
  + silent: <val>
    - Optional. Replace <val> with **true** to not show a message saying you decreased the bond with the Social Link. You could then add your own unique message in an event before or after you call this function.
    - If this isn’t set, it will show a message as part of the function.
* Alternate function: pbLowerSLBond
* Examples:
  + pbLowerSocialLinkBond(:JOEY) => Decreases the bond with :JOEY by 1.
  + pbLowerSLBond(:PROFOAK, 2, silent: true) => Decreases the bond with :PROFOAK by 2, but does so without including a message saying so.

pbSetSocialLinkBond(SocialLinkID, Value, silent: <val>)

* Description: Sets the bond with specified Social Link to a specific value. Returns **true** if the bond was successfully changed. Returns **false** if it was not changed.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to change the bond with.
  + Value
    - Required. Set to an integer representing what you want to set the bond to.
  + silent: <val>
    - Optional. Replace <val> with **true** to not show a message saying you changed the bond with the Social Link. You could then add your own unique message in an event before or after you call this function.
    - If this isn’t set, it will show a message as part of the function.
* Alternate function: pbSetSLBond
* Examples:
  + pbSetSocialLinkBond(:JOEY, 9) => Sets the bond with :JOEY to 9.
  + pbSetSLBond(:PROFOAK, 5, silent: true) => Sets the bond with :PROFOAK to 5, but does so without including a message saying so.

pbGetSocialLinkBond(SocialLinkID)

* Description: Returns the value of the bond you have with the specified Social Link.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to get the bond with.
* Alternate function: pbGetSLBond
* Examples:
  + pbGetSocialLinkBond(:JOEY) => Returns the value of the bond you have with :JOEY as a Social Link.
  + pbGetSLBond(:PROFOAK) => Returns the value of the bond you have with :PROFOAK as a Social Link.

pbSocialLinkBondMaxed?(SocialLinkID)

* Description: Returns **true** the value of the bond you have with the specified Social Link is maxed. Otherwise, returns **false**.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to check.
* Alternate function: pbSLBondMaxed?
* Examples:
  + pbSocialLinkBondMaxed?(:JOEY) => Returns **true** if the value of the bond you have with :JOEY is maxed.
  + pbSLBondMaxed?(:PROFOAK) => Returns **true** if the value of the bond you have with :PROFOAK is maxed.

pbSetSocialLinkLocation(SocialLinkID, Location)

* Description: Changes the location that appears on the specified Social Link’s social media profile.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to change.
  + Location
    - Required. Set to a String representing the location you want to appear for the Social Link Profile.
* Alternate function: pbSetSLLocation
* Examples:
  + pbSetSocialLinkLocation(:JOEY, \_INTL(“Route 3”)) => Sets the location that appears on the social media profile for :JOEY to “Route 3”.
  + pbSetSLLocation(:PROFOAK, \_INTL(“Indigo League”)) => Sets the location that appears on the social media profile for :PROFOAK to “Indigo League”.

pbSetSocialLinkFavoritePokemon(SocialLinkID, Species, Gender, Form, Shiny)

* Description: Changes the Pokemon that appears on the specified Social Link’s social media profile as their favorite.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to change.
  + Species
    - Optional. Set to a :SYMBOL representing the Pokemon species you want to appear for the Social Link Profile.
    - If this isn’t set, it will remove any existing Pokemon from the profile.
  + Gender
    - Optional. Set to 0 for male, 1 for female.
    - If this isn’t set, it will be randomly selected.
  + Form
    - Optional. Set to the form number you want to use.
    - If this isn’t set, it will default to whatever will naturally generate for your game.
  + Shiny
    - Optional. Set to true or false depending on if you want the Pokemon to appear as shiny or not.
* Alternate function: pbSetSLFavPoke
* Examples:
  + pbSetSocialLinkFavoritePokemon(:JOEY, :RATTATA) => Sets the favorite Pokemon that appears on the social media profile for :JOEY to be Rattata.
  + pbSetSLFavPoke(:PROFOAK, :CASTFORM, 1, 2, true)=> Sets the favorite Pokemon that appears on the social media profile for :PROFOAK to be a shiny, female, form 2 Castform.
  + pbSetSocialLinkFavoritePokemon(:BROCK) => Makes it so the social media profile for :BROCK doesn’t show a Pokemon.

pbSetSocialLinkStatus(SocialLinkID, Status)

* Description: Changes the status that appears on the specified Social Link’s social media profile to be either a specified string, or a status from their defined :static\_status\_pool.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to change.
  + Status
    - Required. You have two options:
      * Set to a String representing the status you want to appear for the Social Link Profile.
      * Set to an Integer representing the index of the status to use from the Social Link’s :static\_status\_pool defined in its configuration.
* Alternate function: pbSetSLStatus
* Examples:
  + pbSetSocialLinkStatus(:JOEY, \_INTL(“Shorts are so comfy!”)) => Sets the status that appears on the social media profile for :JOEY to “Shorts are so comfy!”.
  + pbSetSLStatus(:PROFOAK, \_INTL(“… how do I use this?”)) => Sets the status that appears on the social media profile for :PROFOAK to “… how do I use this?”.
  + pbSetSLStatus(:BROCK, 2) => Sets the status that appears on the social media profile for :BROCK to the status defined in index 2 of :BROCK’s :status\_status\_pool.

pbSetSocialLinkStatusRandom(SocialLinkID)

* Description: Changes the status that appears on the specified Social Link’s social media profile to be a random one from their defined :random\_status\_pool.
* Arguments:
  + SocialLinkID
    - Required. Set to the ID of the Social Link Profile you want to change.
* Alternate function: pbSetSLStatusRandom
* Examples:
  + pbSetSocialLinkStatusRandom(:JOEY) => Sets the status that appears on the social media profile for :JOEY to a random one from the pool of available statuses set in the configuration.

pbSetSocialTheme(Color)

* Description: Changes the theme color of the Social Links app.
* Arguments:
  + Color
    - Required. Set to a string that is the name of a folder in the Graphics > UI > Social Links > Themes folder.
* Example:
  + pbSetSocialTheme(“Green”) => Sets the theme of the Social Links app to be green.

**Manual Setup – Defining your own Social Links**

The core of this plugins involves defining Social Links for your own game. Review the definition instructions in the 002\_Configuration file for how to define these.

**Manual Setup – Implementing Script Variables, EXP Boosts, and Shiny Rate Boosts**

In order to help reduce incompatibilities with other plugins you may use, I decided to not overwrite a few functions to only add a single line to implement Script Variables, EXP Boosts, and Shiny Rate Boosts. Instead, please do the following to enable each feature:

*Script Variables*

1. Open the Game\_Event script in the Script Editor and locate the refresh function. (Note, you may have to edit a plugin instead if one overwrites that function).
2. Find the following line in that script:

next if c.variable\_valid && $game\_variables[c.variable\_id] < c.variable\_value

1. **Comment out** that line, and add the following line:

next if c.variable\_valid && varValue?(c.variable\_id) < c.variable\_value

1. Save

*EXP Boosts*

1. Open the Battle\_EXPAndMoveLearning script in the Script Editor and locate the pbGainExpOne function. (Note, you may have to edit a plugin instead if one overwrites that function).
2. Find the following line in that script:

exp = exp \* 3 / 2 if $bag.has?(:EXPCHARM)

1. **After** that line, and add the following lines:

if bond\_rate = $player.active\_bond\_effect?(:EXP, pkmn)

exp = (exp \* bond\_rate).floor

end

1. Save

*Shiny Rate Boosts*

1. Open the Overworld\_WildEncounters script in the Script Editor and locate the pbGenerateWildPokemon function. (Note, you may have to edit a plugin instead if one overwrites that function).
2. Find the following line in that script:

shiny\_retries += 2 if $bag.has?(:SHINYCHARM)

1. **After** that line, and add the following lines:

if bond\_rate = $player.active\_bond\_effect?(:Shiny, genwildpoke)

shiny\_retries += bond\_rate

end

1. Save