Complete Street Rule Documentation & User Guide

By David Wasserman

December 10, 2020

Chapter Guide

Contents

Complete Street Rule Documentation & User Guide 0

Chapter Guide 2

Summary 4

General Review CityEngine Streets 4

Review of CityEngine Streets and Start Shapes 4

A Note on Street UV Texture Coordinates 4

Street UVs- Sets 1 and 2: Crosswalks 4

Street UV - Set 0: Main Street Section 5

A note on "Space Conflicts" 5

General Architecture 5

Right and Left Sidewalk Start Shapes: 6

1. Left and Right Split Space (Bike lanes, Buffers, and Parking): 6
2. Left and Right Floating Transit lanes: 7
3. Left and Right Main Lanes: 7
4. Center Section Split: 7

Intersections and Other Street Start Shapes: 7

- Freeway and Freeway Entry: 7
- Roundabout and Roundabout Island: 8

Parameter Guide 8

Parameter Group: Display Options 8

Parameter Group: Road Lay Out 10

Basic Components 10

Stop Markings 11

Crosswalk Markings 11

On-Street Parking 13

Parameter Group: Center Section Layout 15

Basic Attributes 15

Median Plantings 17

Basic Components 19

Parameter Group: Multimodal Lanes Layout 20

Bus and HOV Lanes Layout 20

Bike Lanes 21

Bike Box ... 23
Parameter Group: Sidewalk Layout ... 24
Sidewalk Attributes ... 24
Sidewalk Plantings ... 24
Sidewalk Components ... 27
Parameter Group: Population ... 28
Parameter Group: Custom Objects ... 29
Parameter Group: Bridges ... 29
Parameter Group: Trees (Imported from ESRI.lib) ... 30
Reports Discussion ... 31
Arbitrary Geometry Reports ... 31
Braking Sight Distances ... 32
Multimodal Segment Evaluations ... 33
Mode Area ... 33
Curbside Allocation ... 34
Handles ... 34
Dashboards ... 35
Mode Preference Dashboard Examples: ... 36
Speed and Stopping Distances Dashboards Examples: ... 37
Mode Area Dashboards Examples: ... 37
Curbside Allocation Dashboards ... 38
Street Trick Notes ... 38
Cross-Section Creation and Image Taking ... 38
Creating Curb Extensions: ... 42
Other Forum Discoveries: ... 42
Removed and Deprecated Features Archive ... 42
Paint Reports ... 42
Paint Reporting ... 43
Object Counts ... 43
Removed Parameters ... 44
Questions and Feedback: ... 44
Feature Requests ... 44

Summary

This is general guide to the complete street rule is an attempt to provide some basic documentation about the Complete Street Rule's general design, parameters, reporting aspects, and some tricks with the rule. There is more discussion about the Complete Street rule on this discussion page on Geonet, but many of the main aspects of this discussion page will be relayed in this document. This guide has since been updated to include documentation about the Complete Street Rule Update, and some of the new features it includes. The page where the new rule is hosted is being updated regularly with changes, so any differences between the old a new rule will be most apparent on the hosting page, but key changes will be denoted with a note that says Updated Rule Change in the parameter guide. In addition, a small short chapter about dash boards and reporting changes will be included.

Dependencies:

The complete street rule uses aspects of CityEngine 2014 streetUVs, and uses the Tree Rule in ESRI.lib (so it only works on CE Licenses 2014 and up). The core assets required for the Complete Street rule to work are the CGA rule, and the Assets of the complete street rule on the **top level** of your projects assets folder.

General Review CityEngine Streets

Below what follows is a review of Street Start Shapes, A note on StreetUVs, and then a description of the general design of the Complete Street rule.

Review of CityEngine Streets and Start Shapes

All street segments in Esri CityEngine have 3 basic start shapes. A right and left sidewalk, and a street start shape that are core components of the Complete Street Rule. Start shapes have their own "Start Shape Parameters" which are attributes that are inherent to that start shape, and in the case of streets the Street shape parameters govern the core attributes that exist for each street segment. Along with street segments, intersections (nodes) also have their own start shapes and parameters that have their own unique attributes that manifest themselves depending on the orientation and angle of intersection with other streets. The street segments in CityEngine have the ability to be "Smart" in the sense that they can for example change an intersection with a "Crossing" start shape that is best for crosswalks and urban street intersections, become a highway merging lane when the angle of intersection is acute enough. The CityEngine documentation can go into more detail about street start shapes and how streets work in CityEngine generally, but a core thing to remember when working with the complete street rule is that start shape parameters **are not** rule parameters. What this means is that start shape parameters are an inherent property of each respective shape "class" such as streets and intersections, and are not parameters that change from rule to rule that are applied to them. Keep this mind while reading the rest of this guide.

A Note on Street UV Texture Coordinates

Many of the CGA functions in CityEngine are oriented around texturing 3D models with custom images to make them more realistic. In the case of streets, an understanding of the details of these Street UVs is critical to understanding how street rules work, and understanding the current limitations of the existing street rule. While the documentation goes into detail about how Street UV's work, what follows is a small description of how these UV's relate to the complete street rule and the importance of certain parameters to having a good looking procedural street.

Street UVs- Sets 1 and 2: Crosswalks

The documentation states that "UV Sets 1, and 2 provide distance fields from the start of the entry, and end of the exit respectively." What this means is that the start and end of a street segment that wants crosswalks that tracks corners would be best served to use this UV for crosswalks. This UV setting works great at making sure your crosswalks make sense in context regardless of the angle of intersecting streets. One issue with these UV's however when they meet is or "conflict" with the main streets UV space (Set 0), it can start clipping geometry. More or this below. The crosswalks and stop bar gaps are controlled in the Crosswalks section of the panel. An important thing to understand is that the initial settings for the crosswalks are determined by the [ConnectionEnd and ConnectionStart]() street parameters inherent to streets. These object attributes give some information about what shape the street might be immediately adjacent to. This is used by the rule to create initial settings. More information on these sections are in the parameter guide.

Street UV - Set 0: Main Street Section

The documentation states that "The first UV set provides a set of street lanes." What is important to understand about this UV set for streets is that its splits can be done two ways. The first way is with UnitSpace splits which are done in absolute units of distance (meters), and the others is UVSpace Splits which are normalized by the street parameters "lane width". What is important to know about how this relates to the Complete Street rule is that the complete street rule uses **UnitSpace** for its splits along the **main center region of the road** (Set 0), but uses the **UVSpace Splits** for its **crosswalks (Sets 1 and 2).** The largest reason for this is that the lanes for the street rule had to use absolute geometry, and had to be able to differ across lane type (it does not make sense to depend on a single lane width street parameter provided by the shape when you want custom width transit lanes and bike lanes). While it might be possible to use UVSpace for the entire rule, the problem was much simpler to tackle when the rule was constructed to try to use the best of both worlds when working with the two spaces.

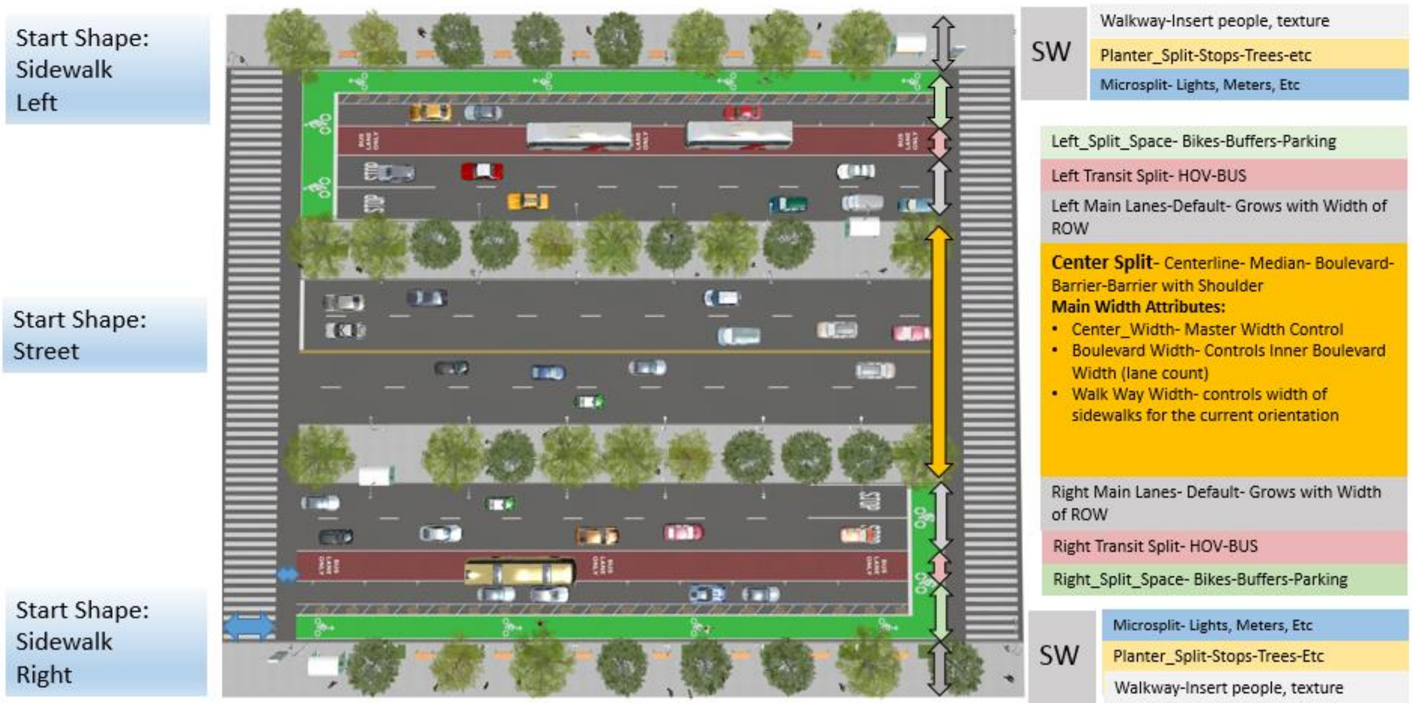
A note on "Space Conflicts"

The issue that comes with this though is that the UVSpace sets 1 and 2 and the UnitSpace in set 0 have the potential to have "conflicts", where generally UV Sets 1 and 2 (Crosswalks) will start eating geometry of UV Set 0 (Main Center Section). The way that this issue was mitigated was by using the space between the crosswalk and the stop bars as "filler" that could be eaten away on angled intersections (intersections that meet at an extreme obtuse or acute angle), without impacting how the street rule functioned. **Something** to be clear about is that if the crosswalk UVs eat too much geometry, the way the street calculates width and other aspects of the rule might change *(harming overall function and stability of the rule).* Thus, it might help to increase the spacing between the stop bars and the crosswalk at weird intersections. In most cases though, this problem is something the user does not have to worry about, especially if they are concerned with mainly cross sections. If you run into trouble with conflicts, try **adjusting the End/Begin_Crosswalk_To_Stop_Bar parameters till your street works with its current orientation/configuration.**

General Architecture

The Diagram above represents a schematic representation of the main splits within the complete street rule. It is intended to be a reference throughout the rest of the documentation, and aid in any customization efforts. The term **"split" or "splits" (plural)** that is used throughout this document refers to the [split operation]() in CGA, which is used to subdivide street shapes within the street rule. Other operations will be discussed when relevant, but the split operation is important in that it heavily depended on to organize the structure of the rule.

General Architecture: Main Splits



Right and Left Sidewalk Start Shapes:

The sidewalks in CityEngine are governed by the Right and Left sidewalk width parameters in the street parameters. The sidewalk start shapes have 3 main splits in the original geometry that is generated from.

1. **Sidewalk Furniture Zone Width:** The first split closest to the road, is a microsplit whose width is determined by the by an attribute that that is set to .2 m (about half a foot) by default. This space is reserved to host parking meters, street lamps, and other curb side assets. The top texture for this space is the same as the sidewalk texture. As of 2018, this is an adjustable width setting for sidewalks.
2. **Planter Split:** The second horizontal split is where the planters and benches are placed. This is defined by the planters width attribute.
3. **Walkway:** The last split represents the space for the walkway of the sidewalk. People are inserted here along with the sidewalk texture. It is the remainder of the furniture zone width and the planters width attribute.

In the vertical direction, there is a “floating split” that governs the insertion of bus stops. This split only allocates geometry if a bus stop attributes are not set to none, thus it is floating in the sense it is not really noticeable unless a bus stop is turned on. Other vertical splits exist for much of the object insertion and creating of plantings. Street Start Shape:

The street start shape is where most of the code of the rule resides, and thus has a complex assortment of layered splits. However the rule’s main splits can be broken down into 4 main types, and how they operate within the rule is discussed below.

1. Left and Right Split Space (Bike lanes, Buffers, and Parking):

This main rule allocates geometry for three separate street lane allocations: bike lanes, bicycle buffers, and on street parking. The reason why these three lane allocations are grouped is that they have some level of interdependence. The bicycle rule attributes have a Buffer Protection and a Parking Protection attribute that are arranged to flip both the buffer and the parking lanes position relative to the bike lanes. The way this is done is by creating two layers for the three lane types, and then using scale UV operations to flip them. In addition to changing orientation these flipping operations also change the reporting for bicycle stress by changing the reporting level of separation. There are right and left side version of these rules.

2. Left and Right Floating Transit lanes:

These rules are referred to as floating rules, because there are technically 3 splits that could all potentially allocate geometry for the transit lanes. This was done so that multiple locations could potentially support transit lanes including the right most lane, left most lane, and a curbside lane. The reason for this complex allocation mechanism for the transit lanes, is that in order to have a variable width transit lane, there had to be three separate allocation positions for potential transit lane positions. So these transit lanes are floating in the sense, that the splits are only allocated geometry IF the relevant lane options is chosen. There are left and right side versions of these rules.

3. Left and Right Main Lanes:

These rules are designed to create the main travel lanes in the street rule. The Main lane rules are designed to consume any geometry that is not allocated within the current street width, but only if there is enough space to create a lane equal to the current Lane Width (set by the relevant attribute). What this means is that rounding error is possible within the rule in the sense that you can have up to a max of a little less than ½ a Lane Width of unallocated "empty space" at any one time. The rule simply allocates lane texturing with a repeating lane texture. The stop textures are determined by 14 m splits before the stop bar, and determined based on the relevant attributes. There are left and right side versions of these rules.

4. Center Section Split:

The center section refers to a large collection of rules that govern how a split in the center of the street segment is allocated. This split follows a large series of case logic that designates this section different rules to allocate geometry based on what center type attributes are selected. In the case of Barriers, Barriers with Shoulders, Medians, and Center Turn Lanes the geometry allocation is pretty apparent in that the larger the center width, the large the section being designated the main geometry. However, the case is more complex with boulevards where there are three main internal splits applied to the Center Section (2 side medians, and a center lane section for Bus Lanes or Travel Lanes) thus has more attributes controlling it. There is only one center section, and the center section options tend to balance geometry between the right and left side of the center section equally.

Intersections and Other Street Start Shapes:

In most cases, any shape that is not a street or sidewalk start shape are given less attention, and are usually sent to the asphalt rule (thus will show asphalt texturing at crossings, junctions, and joints). All of the start shapes are given area reporting appropriate to their start shape ("Crossing Area", "Freeway Area"), but also are sent to the BridgeMain Rule just as the Sidewalks and Street Start shapes are (creates the bridges for the streets-By: Chris Wilkins). However the start shapes listed below are the exceptions to this rule.

- Freeway and Freeway Entry: These rules are designed to create freeway entry points and freeway lanes when the segment line angle is acute enough. The rules should show lanes gradually merging into a freeway start shape. Much of this rule is just a copying of the rule patterns of Street_Modern_Standard.CGA rule created by ESRI Zurich. This rule also has its own reporting and a call to the BridgeMain rule.

Figure 1 Freeway Example

- Roundabout and Roundabout Island: The roundabout start shape defaults to be sent to plain asphalt, but the rule also has basic support for roundabout islands. If you select the roundabout start shape and change the Sidewalk_Ground_Cover option to anything that is not “None” will create a grass covered Roundabout Island with trees. Other than the random scattering of trees (reacts to Sidewalk_Tree_Percentage), and the creation of an island, there is not much done with this start shape. This rule also has its own reporting and a call to the BridgeMain rule.

Figure 2 Roundabout Island Example

Parameter Guide

The parameter guide that follows provides basic descriptions of each of the Street Rules parameters. In cases where parameters are very similar in function, the entire group will be given a description rather than each individual parameter. Keep in mind that numbers are in *meters* for the defaults. In addition, keep in mind that these parameters are inherent to the Complete Street Rule, and not a parameter of a specific shape type such as “Streets” and “Joints”.

Parameter Name	Description
Parameter Group: Display Options	
Display_Textures	Default Value: true Description: When true, textures are on display, when false textures are removed. If true it deletes an unused UVset (see notes for details). Notes: The way the Display Textures setting works is honestly somewhat of a clever hack. There is a deleteUV operation on every texture projecting rule, and when this is set to true, it deletes UV channel 4 (opacity map channel, not used in rule), and when it is false, it deletes UV channel 0 (colormap) which will remove all textures projected on the street. Other methods shown in other street rules had issues actually with the bicyclists and other lanes texturing. This

	solution was actually found to be the easiest and accidentally stable solution by the author, regardless of how much of a hack it is…I would do it again.
Display_Thematics	Default Value: "Thematics Off" Description: Visually colors the entire street model based on the attributes of a street. Usage thematic looks best when textures are turned off. Notes: The thematic are essentially colorings of the street which vary based on the option selected. Solid color allows you to color a street a custom color for some type of emphasis, the stress metrics will color the segments based on the attributes they have and how they play into the different metrics, and the Usage thematic colors each lane designation based on the user it is allocated to (Green for bikes, red for transit lanes, white for sidewalks etc).
Solid_Color	Default Value: "#FFFFFF" Description: When the Solid Color thematic is used for highlighting certain streets, this chooses the color that is utilized by the thematic. Notes: Default value is a hex code, but when selected in the inspector you can pick a custom color because this attribute has the Color Annotation.
Space Management: *(Updated Rule Only)*	Default Value: "Best Fit" Description: "Determines whether the lane width will be the best of fit, exact width, or flag unallocated space red when it is 1/3 the current lane width (usually 3-4 feet). Notes: This attribute drastically improves the usability of the rule by allowing lanes to adjust in size to consume empty space if it is set to best fit, if it is set to exact, the cross section is exact geometry again, and the flag setting will make empty space red if it is greater than 1/3 the current lane width. This functionality exists by making the lane grow if there is empty space, so this means that the lane width IS the effective minimum width of the street (so this setting will not tell you there is more space for improvements than there really is).
LOD_Setting	Default Value: "high" Updated Default: **"Moderate"** Description: This attribute controls the level of detail of the selected textures and OBJs. Typically a lower LOD will decrease the polygon count. Notes: For more polygon reduction also switch to trees to fans when this is set to true. Updated Rule Notes**: In the updated rule, LOD has 3 main settings:** **Very High: Uses high detail object, splits, and populates the rule.** **High: Adds uses high detail objects and splits.** **Moderate: No cars or population but uses high LOD assets.** **Low: Uses low LOD assets, no cars or population, and uses different component splits to reduce polygon count.**
Transparency	Default Value: 0 Description: Controls the transparency of models created by the rule. Notes: Controls all inserted models transparency as well.

Parameter Group: Road Lay Out

Basic Components

Lane_Distribution	Default Value: .5 Description: Represents the fraction of the lanes allocated to the *Right Lanes*. If 1 or 0, they become 1 way streets. Notes: This attribute is used to determine how many lanes are allocated to the right side, with .5 referring to an even distribution of lanes. This makes the lane allocation mechanisms more robust than a number of lanes solution because it can fit any arbitrary number of lanes. It is possible to modify the rule to fit a number of lanes solutions however. A neat trick with this is that if you set the lane distribution to .01 or .99 you maintain the yellow centerline which can help with making contraflow lane types.
Lane_Width	Default Value: (laneWidth-.2) Description: Determines the widths of the main travel lanes. Typically, Freeways are about 12 feet (3.6 m), Arterials 11-12 feet (3.3-3.6 m), Collectors 10-12 feet (3.0-3.6 m), and Local roads 9-12 feet (2.7-3.6 m). Notes: This attribute controls the widths of the main travel lanes. The current default value was determined so that the default width would be around 11 feet because default laneWidth (a street parameter attributes) is around 3.5 for street shapes. The reason why the default value is the (laneWidth-.2), is related to how the street rule deals with the geometry allocation from the centerline. If just the street parameters lane width is used, too much geometry is allocated to lanes, and not enough is allocated to the yellow center lines that appear by default.
Traffic_Direction	Default Value: "right-hand" Description: Orients the road for right vs. left traffic. Please note that some aspects of the rule do not *flip* when this is changed such as the median. Notes: This attribute determines the orientation of the roads (right vs. left sided traffic. This is largely an option that has been debugged thoroughly because active efforts of Alan Klys on Geonet.
Speed_Limit_in_MPH	Default Value: _InititalSpeedLimit Description: A descriptive attribute that feeds into reporting. If >= 40 mph, Design Speed is calculated as Speed_Limit +7.5, if less than 40, design speed is assumed equal to the Speed_Limit (see comments for details). 1 MPH==1.6093 KPH. Notes: This attribute represents the posted speed for the street, and feeds largely into reporting for various attributes. However, based on this attributes there is a function that calculates design speed (used for Braking sight distances). The design speed is abstracted away from the user to reduce attribute clutter, and the decision about how to determine design speed is discussed in the rule. There is a large comment discussion in the rule about why this attribute was made like this, but I will relay it here: *# Source 8: TRB study on Design vs Speed limits: (Discussion informed by research "Factors used to select design speed are functional classification, rural versus urban, and terrain (used by AASHTO); AASHTO Green Book procedure, legal speed limit, legal speed limit plus a value (e.g., 5 or 10 mph [8.1 to 16.1 km/h]), anticipated volume, anticipated operating speed, development, costs, and*

	consistency (state DOTs); and anticipated operating speed and feedback loop (international practices)." *#While the profession has a goal to set posted speed limits near the 85th percentile speed (and surveys say that 85th percentile speed is used to set speed limits), in reality, most sites are set at less than the measured 85th percentile speed.* *#NACTO has different design philosophy. "To counteract these gruesome and unnecessary injuries and fatalities, cities should utilize speed control mechanisms that influence behavior, lower speeds, and in turn, reduce injuries and fatalities. Embracing a proactive design approach on new and existing streets with the goal of reducing speeds "may be the single most consequential intervention in reducing pedestrian injury and fatality. Thus they put forward: "Proactive Urban Street Design: Target Speed = Design Speed = Posted Speed"* *As a result of existing research relating to traffic speed and crash risk, and the fact that for the most part this rule is designed for urban streets, if the speed limit is greater than or equal to 40 mph (5 above NACTO suggested speed). The design speed is assumed to be Speed_Limit +7.5 otherwise it is the Speed_Limit- a decision between the two philosophies.*
Stop Markings	
Stop_Begin	Default Value: _getInitialStop(connectionStart,nLanesLeft+_Lt_Transit_Lane_Count) Description: "The initial stop markings do not take into account the topology of the intersection i.e. they need to be set manually." Notes: The stop markings attributes refer to the right and left stop markings that are textured near intersections. The way the lane markings are coded is set so that it can adjust to an arbitrary number of lanes, but looks best when there are no more than 5 in either direction for some of the settings (arrows all lanes for example). The function that chooses the default stop marking stochastically, favoring different types based on the number of lanes or connection type. Generally though a stop type that is not "None" is allocated based on whether the connectionStart street parameter is Crossing, Junction, Roundabout, or Junction Entry AND the street is longer than 30 meters (about 90 ft.).
Stop_End	Default Value: _getInitialStop(connectionEnd, Distribute_Right_Lanes+_Rt_Transit_Lane_Count) Description: "The initial stop markings do not take into account the topology of the intersection i.e. they need to be set manually." Notes: The stop markings attributes refer to the right and left stop markings that are textured near intersections. The way the lane markings are coded is set so that it can adjust to an arbitrary number of lanes, but looks best when there are no more than 5 in either direction for some of the settings (arrows all lanes for example). The function that chooses the default stop marking stochastically, favoring different types based on the number of lanes or connection type. Generally though a stop type that is not "None" is allocated based on whether the connectionEnd street parameter is Crossing, Junction, Roundabout, or Junction Entry AND the street is longer than 30 meters (about 90 ft.).
Crosswalk Markings	
Crosswalk_Begin	Default Value: _getInitialCrosswalk(connectionStart) Description: NACTO-High--visibility ladder, zebra, and continental crosswalk markings are preferable to standard parallel or dashed pavement markings. These are more visible to approaching vehicles and have been shown to improve yielding behavior. Notes:

	This attribute controls the crosswalk type that will be generated on a street segment by the rule, and its initial settings are determines by the connectionStart street parameter. Generally this attribute's default setting is set to create a continental a crosswalk that is usually continental (if _PedRank>.5 it is always continental) IF the connection start is equal to a Crossing, Junction, Roundabout, or Junction Entry. If those start shapes are not the ones it connects to (Freeways or None) it will be set to No Crosswalk. Crosswalks always look best when the intersection is set to the Crossing start shape because it is more compact than the junction and tends to preserve geometry more effectively.
Crosswalk_End	Default Value: _getInitialCrosswalk(connectionEnd) Description: NACTO-High--visibility ladder, zebra, and continental crosswalk markings are preferable to standard parallel or dashed pavement markings. These are more visible to approaching vehicles and have been shown to improve yielding behavior. Notes: This attribute controls the crosswalk type that will be generated on a street segment by the rule, and its initial settings are determines by the connectionStart street parameter. Generally this attribute's default setting is set to create a continental a crosswalk that is usually continental (if _PedRank>.5 it is always continental) IF the connection start is equal to a Crossing, Junction, Roundabout, or Junction Entry. If those start shapes are not the ones it connects to (Freeways or None) it will be set to No Crosswalk. Crosswalks always look best when the intersection is set to the Crossing start shape because it is more compact than the junction and tends to preserve geometry more effectively.
Begin_Crosswalk_To _Stop_Bar	Default Value: _getInitialGap(connectionStart) Description: Crosswalk to Stop Bar Distance. If the Crosswalk UV is clipping geometry, adjust this to set it back more on angled streets. If used, stop and yield lines should be placed a minimum of 4 ft (1.2 m) from the Crosswalk-MUTCD. NACTO suggests a minimum of 8 ft (2.44 m) in urban areas. Notes: This attributes controls the distance between the stop bar and the crosswalk of the relevant side. As noted in UV Space conflicts, clipping can occur on some sides of the street depending on how the street meets the next intersection (number of streets intersecting and angle of intersection play a role). Increasing the distance helps deal with weird and errant streets.
End_Crosswalk_To_Stop_Bar	Default Value: _getInitialGap(connectionEnd) Description: Crosswalk to Stop Bar Distance. If the Crosswalk UV is clipping geometry, adjust this to set it back more on angled streets. If used, stop and yield lines should be placed a minimum of 4 ft (1.2 m) from the Crosswalk-MUTCD. NACTO suggests a minimum of 8 ft (2.44 m) in urban areas. Notes: This attributes controls the distance between the stop bar and the crosswalk of the relevant side. As noted in UV Space conflicts, clipping can occur on some sides of the street depending on how the street meets the next intersection (number of streets intersecting and angle of intersection play a role). Increasing the distance helps deal with weird and errant streets.
Crosswalk_Color	Default Value: "white" Description: Determines the color of painted crosswalks.

	Notes: There are only two options for this at the moment, white and yellow, but this only applies to painted crosswalks.
Custom_Crosswalk_ Texture	Default Value: SidewalkFolder+"/Paver Brick Red Basket Weave.jpg" Description: Creates a 2 m by 2 m tile of the image selected on the crosswalk. Keep in mind that if another painted tile is chosen, the paint area will not appear in reporting. Defer to Crosswalk Area Reports. Notes: This allows custom sidewalk textures by a user, when the custom options are chosen. To use a picture you have but is not included in the assets folder, open up the assets folder and put the picture you want within it (preferably the sidewalk folder). Then when you try to change this parameter a file picker will appear for you to use.
Crosswalk_Width	Default Value: _crosswalkWidth Description: This attribute overrides the default Crosswalk Width. NACTO-Stripe the crosswalk as wide as or wider than the walkway it connects to. Notes: So the default setting for the crosswalk is to be the width of the max of right or left sidewalk width +.05 m, but if the width is 0 for both of them for some reason it defaults to 3 (because other options above will just turn off crosswalks if the user turned them off, so it is assumed that a crosswalk of some type is desired. Again this default setting can and should be overridden if desired.
On-Street Parking	
Right_Parking_Type	Default Value: "None" Description: Will create parking of that type with default lengths and widths. Keep in mind, adjusting the length/width will make it lock onto that value. To reset it to automatic default, set the Attribute Connection Editor back to rule-defined value. Notes: This attribute applies to the Right Side of the street (orientation shape dependent). What is important to know is that this attribute controls the appearance of parking lanes on the street, and that it will give them default values informed by AASHTO and other Guidelines. However, once a user changes the default for this parameter, the width and length become locked at that value. Recent versions of this parameter now support additional allocations for curbside features other than parking lanes.
Right_Parking_Width	Default Value: _ParkingWidth("Right") Description: Good default is 8 feet (2.4384 m) with a minimum of 7 feet (2.1336 m) for low turnover locations for parallel parking. Angled parking varies on angle, but a 30-45 degree depth suggestion is 19 feet (5.7912 m). Design guidance on width varies with conditions. Notes: Attribute overrides default widths set by the Parking Width function. The worker function operates to determine the default width based parking type set and the side of the street it is on. General defaults follow what is shown in the description.
Right_Parking_ Length	Default Value: _ParkingLength("Right") Description: For Parallel Parking 6.1 to 7.5 meters long is suggested, for Angled Parking 2.4 to 3 m is suggested.

	Notes: Attribute overrides default lengths set by the Parking Length function. The worker function operates to determine the default length based parking type set and the side of the street it is on. General defaults follow what is suggested in the description. Parking Length generally tends to have less guidance than width generally (AASHTO mentions it briefly), but for this rule length and width have some guidance from this design code here.
Left_Parking_Type	Default Value: "None" Description: Will create parking of that type with default lengths and widths. Keep in mind, adjusting the length/width will make it lock onto that value. To reset it to automatic default, set the Attribute Connection Editor back to rule-defined value. Notes: This attribute applies to the Left side of the street (orientation shape dependent). What is important to know is that this attribute controls the appearance of parking lanes on the street, and that it will give them default values informed by AASHTO and other Guidelines. However, once a user changes the default for this parameter, the width and length become locked at that value. Recent versions of this parameter now support additional allocations for curbside features other than parking lanes.
Left_Parking_Width	Default Value: _ParkingWidth("Left") Description: Good default is 8 feet (2.4384 m) with a minimum of 7 feet (2.1336 m) for low turnover locations for parallel parking. Angled parking varies on angle, but a 30-45 degree depth suggestion is 19 feet (5.7912 m). Design guidance on width varies with conditions. Notes: Attribute overrides default widths set by the Parking Width function. The worker function operates to determine the default width based parking type set and the side of the street it is on. General defaults follow what is shown in the description.
Left_Parking_ Length	Default Value: _ParkingLength(“Left”) Description: For Parallel Parking 6.1 to 7.5 meters long is suggested, for Angled Parking 2.4 to 3 m is suggested. Notes: Attribute overrides default lengths set by the Parking Length function. The worker function operates to determine the default length based parking type set and the side of the street it is on. General defaults follow what is suggested in the description. Parking Length generally tends to have less guidance than width generally (AASHTO mentions it briefly), but for this rule length and width have some guidance from this design code here.
P*arklet_Percentage*	Default Value: 0 Description: Will create Parklets in Parking spaces, it is best if the parking spaces are contiguous to the sidewalk. The default OBJ does not have bollards or curb stops suggested by NACTO. Notes: The Parklet percentage only matters when there is on-street parking (working best exclusively with parallel parking), and is set up to create a parklet that is sized to be the same as the current parking space + empty space that results

	from left over geometry not allocated (meaning that the parklets should ALWAYS be flushed with the curb).
Front_Parking_Spacing	Hidden Attribute (Reveal by Deleting @Hidden Annotation) Default Value: 0 Description: This hidden attribute controls the spacing the parking areas have before the current directions stop bar. Notes: Gives additional control over how far cars are from a certain end of the intersection. Revealing this attribute is an option that can help gain more procedural control of the on street parking.
Rear_Parking_Spacing	Hidden Attribute (Reveal by Deleting @Hidden Annotation) Default Value: 0 Description: This hidden attribute controls the spacing the parking areas have after the crosswalk at the start of the street in the current direction. Notes: Gives additional control over how far cars are from a certain end of the intersection. Revealing this attribute is an option that can help gain more procedural control of the on-street parking.
Curbside_Management_Position	Default: End Spaces (Varies- street length dependent). Description: Describes where on the street reallocated street space is used in parking lanes for parallel parking with curbside management options. Notes: This is a new parameter from 2018 Edits. These features are still in development, but essentially the road map for the rule is support emerging mobility trends related to Curbside Management (TNCs, Bus Stops, Bike Corrales for Scooters/Dockless Bike Share/Docked Bike Share, and Autonomous Vehicles). As it stands, there will more update to parallel parking to be higher level of detail and be flexible enough to adjust to different curb space allocations.
Right_Curbside_Allocation_Length *@Handle Enabled.*	Default: Varies Description: This parameter represents the size of a curbside allocation on the right side of the street. Notes: Handles are visible only when certain curbside features are present.
Left_Curbside_Allocation_Length *@Handle Enabled.*	Default: Varies Description: This parameter represents the size of a curbside allocation on the left side of the street. Notes: Handles are visible only when certain curbside features are present.
Parameter Group: Center Section Layout	
Basic Attributes	
Center_Type	Default Value: Yellow Centerline Description: This attribute is key to picking a center type. If none, it is a centerline, but each choice creates different center section layouts. Notes: This is the most critical control attribute for the Center Section of the street. The default of "Centerline" settings allocates about .4 m (16 inches) to the center section split to create a yellow centerline. All the other allow the creation of a wide variety of center types such as barriers, barriers with

	shoulders, center turn lanes, medians, and boulevards (which are implemented as two medians with some roadway type in the middle). This one attributes controls a lot of the aspects of the center, and depending on the choices here the attributes discussed below can either be incredibly important or irrelevant to the option chosen. For example, if barrier is chosen none of the other center attribute types matter because a default center width is held constant (so a barrier will always take up the same amount of space).
Center_Width	Default Value: (Varies by Center Type Selection) Description: Is the combined center section width regardless of Center type. Does nothing for the barrier selection. Notes: This attribute controls the "Master Width" of the center section split. Without any geometry being allocated here, most center section types will show nothing. Something that is important to consider when using this attribute is that if you don't allocate enough geometry for elements such as boulevards and medians, when the widths of other elements such as the Boulevard_Inside_Width is too long.
WalkWay_Width	Default Value: 0 Description: Is the width of the walkways created by Median. They will override the planting width, and keep in mind there are two walkways for Plant:Walk:Plant. Notes: Each walkway that appears will have a width set by this attribute, however one that is not immediately apparent is that the Plant:Walk:Plant setting of the Planting and Walkway Layout attribute has two walkways (Right and Left Side), so the total walkway is actually double what this width is set to. The reason why this was done was because it was easier to implement at the time (helped with bench insertion code).
Planting_and_Walkway_Layout	Default Value: "Walk:Plant:Walk" Description: Lays out the median/boulevard walkway configurations. In the case of Plant:Walk:Plant-the walkway rule is used twice so adjust walkway width accordingly. Notes: Generally this attribute is used to determine the general layout of the plantings within a median with a walkway. For example, Walk:Plant means a walkway on one side and planting on the other (two splits), while Walk:Plant:Walk has a planting placed in the middle of two walkways (three splits). Plant:Walk:Plant is technically 4 splits because it has 2 walkway splits in the center.
Boulevard_Inside_Width	Default Value: 7.1 Description: In the case of a boulevard, this is the combined width for the lanes and center between the walkways. Each lane set gets half allocated space. Keep in mind this means there is forced symmetry in lanes. Notes: This is really only relevant when the Boulevard is turned on, and it controls how much width is available for the lanes within the boulevard. However, unlike travel lanes for the rest of the rule, the boulevard's lanes force symmetry (as in both directions must have the same number of lanes). Also because the lane

	width is the same as the main travel lanes lane width, any extra space will be allocated with drainage/asphalt geometry. This was largely done to keep the boulevard code as simple as possible.
Boulevard_Configuration	Default Value: "Normal Lanes" Description: Determines the general configurations of the inside of the boulevard. Normal lanes will be equal to the general Lane_Width, and any leftover geometry is filled with drainage filler. Notes: Normal lanes will keep creating lanes if they are given room, however all the other options will fill up the boulevard by either filling it up with plantings or just giving the transit and bike lane options more space. The bike lane option for the boulevard does not impact the Bike Stress reporting, so it will not be included in the stress metrics ranking.
Boulevard_Center_Type	Default Value: "Center Line" Description: Will create the exact center of the lanes in the boulevard. To remove it set the width ==0. Notes: The inside lanes for the boulevard can support a center space that can be a centerline, flexible posts, fences, and even a small median. These are very simple rules that are allocating this geometry, so they don't allocate people or other objects in their centers.
Boulevard_Center_Width	Default Value: **case** *Boulevard_Configuration*=="Normal Lanes": *PaintLineWidth**4 **else**: 0 Description: Determines the width of the center of the Boulevard lanes. Notes: Its default value will be equal to the typical centerline width if Normal Lanes is active, but it defaults to 0 if not. You can change this value if you want to have flexible posts or centerline separating your bus lanes for example.
Median Plantings	
Median_Ground_Cover	Default Value: "Standard Grass" Description: Chooses the grass texture for the planting locations within the Median. Notes: This allows the user to pick from several grass textures for the median locations. An area with a grass texture (either on the sidewalk or the median) will feed into the pervious area reporting for the rule.
Median_Planting_Length	Default Value: 4 Description: Is the approximate length of the green space accommodating trees, it can be used to space out trees more without walkway spacing. Notes: This attribute controls the absolute size (no ' or ~) of the planting space that will hold trees or plantings to be inserted. Making this attribute larger will create fewer trees by increasing the length of the plantings that will hold each tree. Keep in mind that this shape is not part of the walkways.
Median_Tree_Spacing	Default Value: 3 Description: Creates a walkway spacing between created trees. Notes: This attribute controls best of fit length (~) of the cement spaces between trees. Making this attribute larger will create fewer trees by increasing the distance

	between planting areas that can hold trees. Keep in mind that this shape is not part of the walkways.
Median_Tree_1_Type	Default Value: "Random" Description: Determines the species of the tree/plant selected for Tree 1. Random picks from 5 common tree types and is a good default. Notes: **When the street start shape** is selected, changing this attribute will change the **tree model** (not the tree size or radius- See Tree Parameters below) chosen for allocation within the median.
Median_Tree_1_Percentage	Default Value: 1 Description: Reduces the probability of Tree 1 appearing in a typical designated location. Notes: **When the street start shape** is selected, think of this attribute as the percent chance that a Tree 1 (Tree_1_Type), will be inserted in the **median.** Reducing this to 0 will for example leave no trees of Type 1 to be placed. However, this attribute interacts with the Tree_2_Type when it is not set to none, by inserting a tree of Tree_2_Type when it does not insert a Tree_1_Type. For example, if this attribute is set to .5, and Tree_1_Type is a Rose and Tree_2_Type is an orchid then around 50% of the inserted plantings will be roses and the other half will be orchids in the planting areas.
Median_Tree_2_Type	Default Value: "None" Description: Determines the species of the tree/plant selected for secondary tree for more variation. If this is not None, Tree 2 will appear if Tree 1 does not fire with the current percentage. This does mean that you cannot drop tree density if you alternate trees. Notes: **When the street start shape is selected,** if this attribute is None, it does nothing, however, when it is not and the Median_Tree_1_Percentage is less than 1, any trees that are not Tree_1_Type will be inserted as Tree_2_Type.
Median_Tree_Height_1 *@Handle Enabled.*	Default Value: Varies by Tree Type Description: This parameter controls the height of trees of tree type 1 on sidewalks. It will control trees on the median and a few other locations by default. Notes: **This is a handle enabled attribute so individual trees can have their size adjusted. Radius is automatically adjusted based on Tree Rules.**
Median_Tree_Height_2 *@Handle Enabled.*	Default Value: Varies by Tree Type Description: This parameter controls the height of trees of tree type 2 on sidewalks. It will control trees on the median and a few other locations by default. Notes: **This is a handle enabled attribute so individual trees can have their size adjusted. Radius is automatically adjusted based on Tree Rules.**
Median_Tree_Height_Deviation	Default Value: Varies by LOD – Typically 10% Description: This attribute controls the random deviation that applies to tree heights. As a percent of total height. Notes: This was a parameter added to make the trees inserted look more realistic and varied.

Basic Components	
Median_Bus_Stop	Default Value: "None" Description: Determines whether an object is placed and what side of the street or walkways relevant objects are placed. Notes: Median Bus Stops are created typically by 3 floating splits that will allocate geometry for a bus stop location that would serve the side of the street chosen by this attribute.
Median_Bus_Stop_Location	Default Value: "Far-side" Description: Locates bus stop in the appropriate location. Far-side is right after the last intersection, Mid-Block is in the middle of the street, and Near-side is near to the next intersection. Notes: This is the attribute that along with the "Street Side" controlling attribute, will control the geometry allocation process for the bus stops. The base geometry that is bus stop placed on is a split that is 40 ft. long (12.192 meters) and is the default within the rule.
Median_Bike_Rack	Default Value: false Description: Will create 2 bike racks near a bus stop. Notes: Within the bus stop geometry allocation (40 ft. long split), there are splits that will insert 2 bike racks if this is turned to true and a bus stop is appearing in the rule.
Median_Way_Finder	Default Value: false Description: Will create a WayFinder near a bus stop. Notes: Within the bus stop geometry allocation (40 ft. long split), there are splits that will a way finder if this is turned to true and a bus stop is appearing in the rule.
Median_Benches @Handle Enabled	Default Value: "None" Description: Creates benches on the edges of the walkways of the Median. Notes: This attribute controls the edge of the walkways for which Median Benches will be created.
Median_Bench_Spacing	Default Value: 10 Description: Determines the spacing between each Bench. No shape is created in the sections in between objects. Notes: The larger this number, the fewer median benches will appear in the median walkways as the spacing between them changes.
Median_Street_Lamps	Default Value: "Both" Description: Determines whether an object is placed and what side of the street or walkways relevant objects are placed. Notes: Chooses the side of the streetlamps are put on the median.
Median_Street_Lamp_Spacing	Default Value: 10

	Description: Determines the spacing between each Street Lamp. No shape is created in the sections in between objects. Notes: The larger this number, the fewer streetlamps will appear in the median as their spacing changes.

Parameter Group: Multimodal Lanes Layout

Bus and HOV Lanes Layout	
Transit_Lane	Default Value: "None" Description: The controls for the bus lane also control the HOV lane. They are grouped together because they are both considered: High Capacity Lanes: Notes: The HOV and Bus lanes use the same split to allow their relevant lane textures. They are different rules that occupy the same geometry.
Transit_Lane_Sides	Default Value: _Initial_Transit_Lane_Sides Description: Determines side of street preferential lanes are allocated Notes: If the street is one-way, will make initial bus lane side change accordingly to the appropriate right or left side. Keep in mind that this default "smart" logic is meaningless once the value is changed, so you will have to set it back to "Rule Defined Value" if you want it to automatically adjust the sides of the street based on whether it is a one way or two way street.
Transit_Lane_Width	Default Value: 3.3528 Description: Determines the lane width of transit lanes that are not Dedicated Median Bus Lanes. NACTO suggested width is 11 feet (3.3528 m) minimum for Curb side and Median bus lanes but allows for 10 feet (3.048 m) on Off-set bus lanes. Notes: You can make the transit lanes wider. Future versions of the rule will create a curb lane version of the rule so that you can have a curb lane with different widths.
Transit_Lane_Position	Default Value: Right Most Lane Description: Inserts a transit lane at the location specified. Keep in mind this is an insertion not a lane reallocation. Notes: This attribute controls the floating splits that allocate the transit lanes. These splits allocate geometry in between different splits. The three options are left most lane (adjacent to median), right most lane (right most main lane), sidewalk side (forces the lane to be sidewalk side regardless of the presence of other lane types). The sidewalk side option will place a transit lane adjacent to the curb, and this is done by allocating geometry on the other side the bike lane split. This sidewalk side option may interfere with how the bike box looks (see bike box).
Transit_Symbol_Spacing	Default Value: 24.5 Description: MUTCD- Preferential Lanes: Markings spaced as close as 80 feet (~24.5 m) apart might be appropriate on city streets, while markings spaced as far as 1,000 feet (304.8 m) apart might be appropriate for freeways. Notes:

	This general descript applies to any attribute that gives control over the spacing an arbitrary texture "stamp", such as the words Bus Lane Only or the bicycle symbols. In this case it controls HOV and Bus Lane Stamp Spacing.
Bus_Lane_Color	Default Value: "red" Description: NACTO-Red colored paint should be applied to emphasize the lane and to deter drivers from using it. Red paint has higher installation and maintenance costs but has been shown to deter both unauthorized driving and parking in the bus lane. Notes: Denotes the color of the Bus Lane, and feeds into paint cost reporting. Black has no paint cost reporting done for it other than the white paint for "Bus Lane Only".
Transit_Paint_Line_Sides	Default Value: "Both" Description: Controls the white lines on the sides of preferential lanes. Notes: This attribute controls the number of white lines near the edges of the preferential lanes. They can appear on both sides, one side, and no side. These controls feed into the white paint cost reporting and give the user control over the aesthetics.
Bike Lanes	
Right_Bike_Lane_Width	Default Value: 0 Description: Desirable bike lane width adjacent to a curb face is 6 feet (1.8288m), but rideable surface adjacent to a street edge is 4 feet (1.2192m) with minimum of 3 feet (0.9144m). In areas where illegal parking is an issue, at least 5 feet (1.524m) is suggested. Notes: Width of bike lane on the right side of the street.
Left_Bike_Lane_Width	Default Value: 0 Description: Desirable bike lane width adjacent to a curb face is 6 feet (1.8288m), but rideable surface adjacent to a street edge is 4 feet (1.2192m) with minimum of 3 feet (0.9144m). In areas where illegal parking is an issue, at least 5 feet (1.524m) is suggested. Notes: Width of bike lane on the left side of the street.
Bike_Lane_Type	Default Value: "One-Way" Description: The one-way option uses the full lane width for each lane, while the two-way option will allocate half of the width to each sub-lane. Notes: This attribute splits a bike lane into two directions by splitting it in half and then making two lanes adjacent to one another (each supporting different directions).
Right_Buffer_Width	Default Value: 0 Description: Bicycle Buffers have desired minimums of about 3 feet but should be at least 18 inches wide because it is impractical to mark a zone narrower than that. Notes: Controls the width of the buffers split on the relevant side of the street. It is important to note that shoulders are 8 ft. wide, in case that option is intended to be used on the street segment.

Left_Buffer_Width	Default Value: 0 Description: Bicycle Buffers have desired minimums of about 3 feet but should be at least 18 inches wide because it is impractical to mark a zone narrower than that. Notes: Controls the width of the buffers split on the relevant side of the street. It is important to note that shoulders are 8 ft. wide, in case that option is intended to be used on the street segment.
Buffer_Protection	Default Value: true Description: If on/true, buffer is closer to the through lane, and bicycle lane is protected from through traffic. Notes: This attribute uses the layered splits with scaleUV to achieve the effect displayed by the rule. Both of the protection attributes will alter the bike stress metrics by altering the actual width of the bike lane and its level of protection.
Parking_Protection	Default Value: true Description: If on/true, parking lane is closer to the through lane, and bicycle lane is protected from through traffic. Keep in mind how the door zone influences bicycle lane placement. Notes: This attribute uses the layered splits with scaleUV to achieve the effect displayed by the rule. Both of the protection attributes will alter the bike stress metrics by altering the actual width of the bike lane and its level of protection.
Buffer_Type	Default Value: "Painted Stripes" Description: This attribute controls the bicycle buffer type and form, but also can become plain asphalt or a shoulder. Notes: This buffer can be pretty multifunctional. It can be used to add more width to as curb lane as asphalt, or it can be used to create a wide variety for buffer types for bicycle lanes. These include planters, flexible posts, and various curb types. The buffer options do different things or nothing depending on the buffer type chosen.
Buffer_Object_Spacing	Default Value: _Default_Buffer_Object_Spacing Description: Controls the buffers spacing of objects such as tubular markers, planters, and tree/plantings (Trees/Plants match Sidewalk Plantings if selected). Notes: Default value changes based on what type of object is chosen (so trees might get more spacing than say a flexible post).
Bike_Symbol_Spacing	Default Value: 24.5 Description: MUTCD- Preferential Lanes: Markings spaced as close as 80 feet (~24.5 m) apart might be appropriate on city streets, while markings spaced as far as 1,000 feet (304.8 m) apart might be appropriate for freeways. Notes: This general descript applies to any attribute that gives control over the spacing an arbitrary texture “stamp”, such as the words Bus Lane Only or the bicycle symbols. In this case this controls the bike symbols spacing on the bike lanes.
Bike_Conflict_Spacing	Default Value: 0 Description: Creates conflict spacing made up by asphalt gaps in the bike lane for approaching intersections. Notes:

	This attribute creates the typical conflict spacing you would expect at an intersection. Currently the rule will have the conflict spacing have approximately 1 m gaps between each asphalt gap and paint gap. These gaps might best be used when the street is in thin bands equal to the bike lane width, in order to draw and paint short segments that can cross over intersections.
Bike_Lane_Color	Default Value: “green” Description: Determines the color of the bike lane and bike box. Paint reporting costs adjust based on color choices. Notes: Determines the color of the bike lanes, and only have 4 options. The reason options are limited for things like Bike lane color and Transit lane color is because there are custom textures that are projected for each color, so this attribute only has 4 options as a result.
Bike_Paint_Line_Sides	Default Value: “Both" Description: Controls the white lines on the sides of preferential lanes. Notes: This attribute controls the number of white lines near the edges of the preferential lanes. They can appear on both sides, one side, and no side. These controls feed into the white paint cost reporting and give the user control over the aesthetics.
Level_of_Blockage	Default Value: “Rare” Description: Is a descriptive attribute relating to the probability that a bike lane might be blocked due to unloading or double parking that feeds into Thematic and reporting. Notes: Strictly an attribute that feeds into the bike stress reporting. That is this attribute only function.
Bike Box	
Right_Bike_Box	Default Value: false Description: Right Side Bike Box: NACTO-A bike box is a designated area at the head of a traffic lane at a signalized intersection that provides bicyclists with a safe and visible way to get ahead of queuing traffic during the red signal phase. Works best if bike lane is adjacent to sidewalk. Notes: When true, the rule generates bike boxes by allocating geometry on the appropriate side of the street near the same split as the stop bar. The bike boxes ASSUME that the bike lane is adjacent to the sidewalk curb, and any other orientation will have the lines of the bike box not necessarily line up.
Left_Bike_Box	Default Value: false Description: Left Side Bike Box: NACTO-A bike box is a designated area at the head of a traffic lane at a signalized intersection that provides bicyclists with a safe and visible way to get ahead of queuing traffic during the red signal phase. Works best if bike lane is adjacent to sidewalk. Notes: When true, the rule generates bike boxes by allocating geometry on the appropriate side of the street near the same split as the stop bar. The bike boxes ASSUME that the bike lane is adjacent to the sidewalk curb, and any other orientation will have the lines of the bike box not necessarily line up.
Bike_Box_Symbol_Spacing	Default Value: 5

	Description: Is the best fit spacing between the bicycle symbols on the bike box. Can be increased to reduce the number of symbols created. Notes: Altering this attribute will allow either more or less bike symbols based the width provided. The current width of the spacing between symbols is very wide so that in most cases only one symbol appears.
Bike_Box_Length	Default Value: 4.26 Description: NACTO-A box formed by transverse lines shall be used to hold queuing bicyclists, typically 10-16 feet deep (3.048-4.8768 M). Deeper boxes show less encroachment by motor vehicles. Notes: Controls the depth of the bike box.
Bike_Box_Color_Override	Default Value: *Bike_Lane_Color* Description: By default, the Bike Box will match the color of the bike lane, but this attribute can be adjusted to override that choice. Paint reporting costs adjust based on color choices. Notes: This allows a user to override the color of the bike box for a different color from the bike lane. For example, you can have a black asphalt bike lane, with a green bike box by using this attribute.

Parameter Group: Sidewalk Layout

Sidewalk Attributes

Sidewalk_Texture	Default Value: *Default_Pavement* Description: Provides a file picker for the texture choice. This attribute also controls the curb buffer and median walkway textures. Notes: Allows a user to pick a custom image file to texture the sidewalks. Similar to the crosswalk file picker. **Keep in mind that the attribute changes in the sidewalk group only have an effect when the sidewalk shapes are selected.**
Sidewalk_Texture_Scale	Default Value: 1 Description: Adjust the scale of the sidewalk texture. This attribute also controls the curb buffer and median walkway textures. Notes: Allows a user to adjust the "size" of the image being projected onto the texture.
Sidewalk_Texture_Rotation	Default Value: 0 Description: Adjust the angle of the sidewalk texture. This attribute also controls the curb buffer and median walkway textures. Notes: Allows a user to adjust the rotation of the texture being tiled.
Sidewalk_Height	Default Value: 0.102 Description: Determines the height of the sidewalk. Default is 4 inches tall. This attribute also controls the curb buffer and median walkway textures. Notes: Load bearing sidewalks are usually 6 inches thick, but generally they are 4 inches. The sidewalks are simply extruded and then put through a series of components splits to get this effect.

Sidewalk Plantings

Sidewalk_Ground_Cover	Default Value: "None"

	Description: Chooses the grass texture for the planting locations on the Sidewalk or Center Island. Bus Stops and trees will not generate if this is set to None. Notes: This allows the user to pick from several grass textures for the sidewalk locations. An area with a grass texture (either on the sidewalk or the median) will feed into the pervious area reporting for the rule.
Sidewalk_Furniture_Zone_Width	Default Value:.2 Description: Controls the width of the furniture zone (space between the planting and the curb. Notes: **This parameter a was added to the new rule. It replaced the CurbToSidewalkGap constant. I realize this is not exactly a "Furniture Zone" but for the time being this is the name for this parameter.**
Sidewalk_Planting_Width	Default Value: 1.5 Description: Controls the width of sidewalk planting space, and the setback of the bus stop. Notes: This attribute controls the width of the planting areas on the sidewalk shapes. Unlike the Medians where walkways determine what is left to plantings, Sidewalks have a planting width that determine what is left for sidewalks. The edge of this shape closest to the sidewalk is the edge that the Bus stop is aligned to be inserted to, so the wider this attribute, the further back away from the curb the bus stop will be. It helps to either have a higher bus stop set back or a high planting width to have a well inserted bus stop with the street rule. **The Updated Rule has a dynamic default planting width that will shrink or grow based on how much sidewalk space is available.**
Sidewalk_Planting_Length	Default Value: 5 Description: Is the approximate length of the green space accommodating trees, it can be used to space out trees more without walkway spacing. Notes: This attribute controls the absolute size (no 'or ~) of the planting space that will hold trees or plantings to be inserted. Making this attribute larger will create fewer trees by increasing the length of the plantings that will hold each tree. This planting length for the sidewalks will operate similarly to the Median_Planting_Length.
Sidewalk_Planting_Spacing	Default Value: 5 Description: Creates a walkway spacing between created trees. Keep in mind that the benches are placed in this spacing and must be an appropriate size to accommodate them. Notes: This attribute controls best of fit length (~) of the cement spaces between trees. Making this attribute larger will create fewer trees by increasing the distance between planting areas that can hold trees. Keep in mind that this shape is not part of the walkways. In the case of the sidewalks, the benches get placed in between this shape, so there must be enough room to hold them (2 meters), or they will not appear.
Sidewalk_Tree_1_Type	Default Value: "Random"

	Description: Determines the species of the tree/plant selected for Tree 1. Random picks from 5 common tree types and is a good default. Notes: **When the right or left sidewalk start shape** is selected, changing this attribute will change the **tree model** (not the tree size or radius- See Tree Parameters below) chosen for allocation within the planting split on the sidewalk.
Sidewalk_Tree_1_Percentage	Default Value: 1 Description: Reduces the probability of Tree 1 appearing in a typical designated location, but also controls the Tree quantity at roundabouts. Notes: **When the right or left sidewalk start shape** is selected, think of this attribute as the percent chance that a Tree 1 (Tree_1_Type), will be inserted in the **sidewalk.** Reducing this to 0 will for example leave no trees of Type 1 to be placed. However, this attribute interacts with the Tree_2_Type when it is not set to none, by inserting a tree of Tree_2_Type when it does not insert a Tree_1_Type. For example, if this attribute is set to .5, and Tree_1_Type is a Rose and Tree_2_Type is an orchid then around 50% of the inserted plantings will be roses and the other half will be orchids in the planting areas.
Sidewalk_Tree_2_Type	Default Value: "None" Description: Determines the species of the tree/plant selected for secondary tree for more variation. If this is not None, Tree 2 will appear if Tree 1 does not fire with the current percentage. This does mean that you cannot drop tree density if you alternate trees. Notes: **When the right or left sidewalk start shape** is selected, if this attribute is None, it does nothing, however, when it is not and the Sidewalk_Tree_1_Percentage is less than 1, any trees that are not Tree_1_Type will be inserted as Tree_2_Type.
Sidewalk_Tree_Height_1 *@Handle Enabled.*	Default Value: Varies by Tree Type Description: This parameter controls the height of trees of tree type 1 on sidewalks. It will control trees on the sidewalk shapes and a few other locations by default. Notes: **This is a handle enabled attribute so individual trees can have their size adjusted. Radius is automatically adjusted based on Tree Rules.**
Sidewalk_Tree_Height_2 *@Handle Enabled.*	Default Value: Varies by Tree Type Description: This parameter controls the height of trees of tree type 2 on sidewalks. It will control trees on the sidewalk shapes and a few other locations by default. Notes: **This is a handle enabled attribute so individual trees can have their size adjusted. Radius is automatically adjusted based on Tree Rules.**
Sidewalk_Tree_Height_Deviation	Default Value: Varies by LOD – Typically 10% Description: This attribute controls the random deviation that applies to tree heights. As a percent of total height. Notes: This was a parameter added to make the trees inserted look more realistic and varied.
Sidewalk_Planting_Border	Description: This parameter determines what type of border planting boxes have in inserted locations. Notes:

	Provides more detailed planting boxes.
Sidewalk Components	
Sidewalk_Bus_Stop	Default Value: "None" Description: Determines whether an object is placed and what side of the street or walkways relevant objects are placed. Note: Sidewalk Bus Stops are created typically by 3 floating splits (similar to the Median Bus Stops) that will allocate geometry for a bus stop location that would serve the side of the street chosen by this attribute.
Sidewalk_Bus_Stop_Location	Default Value: “Far-side” Description: Locates bus stop in the appropriate location. Far-side is right after the last intersection, Mid-Block is in the middle of the street, and Near-side is near to the next intersection. Notes: This is the attribute that along with the “Street Side” controlling attribute, will control the geometry allocation process for the bus stops. The base geometry that is bus stop placed on is a split that is 40 ft. long (12.192 meters) and is the default within the rule. Works in a similar fashion to the Median Bus Stop allocator.
Sidewalk_Bus_Stop_Setback	Default Value: 1 Description: As it stands, the Bus Stop will be as far back as the Planting Width, if more adjustment is required, this attribute can be used to set back the Bus Stop further back or move it closer to the curb. Notes: The median bus stops just take up the entire geometry they are placed in, but the sidewalk bus stops have a little bit more control over how they are placed on the sidewalk. This attribute controls a translation horizontally on the sidewalk and can be – or +.
Sidewalk_Benches @Handle Enabled	Default Value: "None" Description: Will place benches in between the spacing between Trees on the sidewalk side chosen. Notes: These benches require a planting space large enough to accommodate me (2 meters), if there is not enough space, the sidewalk benches will not appear.
Parking_Meters	Default Value: "None" Description: Determines whether an object is placed and what side of the street or walkways relevant objects are placed. Notes: In the microsplit on the sidewalks, this attribute will allocate parking meters on the sidewalks of the appropriate side that should be adjacent to the parking spaces. Default meter is a fan file with a low polygon count.
Parking_Meters_Spacing	Default Value: 6.1 Description: Determines the spacing between each parking meter. No shape is created in the sections in between objects. Notes: This attribute controls the spacing between each inserted parking meter. If the number increases, you get fewer parking meters, and if it decreases you get more with them closer together. The default is attempting to achieve a spacing for parallel parking lanes.
Parking_Meters_Setback	Default Value: 9

	Description: This attribute provides a way to adjust the starting location for parking meters so that they align with on-street parking. Notes: This adjusts the starting location for the parking meters in case a weird street geometry has the sidewalk and street parking not necessarily line up with the parking meters.
Sidewalk_Street_Lamps @Handle Enabled	Default Value: "None" Description: Determines whether an object is placed and what side of the street or walkways relevant objects are placed. Notes: In the microsplit on the sidewalks, this attribute will allocate streetlamps on the sidewalks of the appropriate side that go over the streets. **Updated Rules Support @Handles for translation for Street Lamps.**
Sidewalk_Street_Lamp_Spacing	Default Value: 10 Description: Determines the spacing between each Street Lamp. No shape is created in the sections in between objects. Notes: This attribute controls the spacing between each inserted streetlamp. If the number increases, you get fewer streetlamps, and if it decreases you get more with them closer together.
Traffic_Lights @Handle Enabled	Default Value: "None" Description: Determines whether an object is placed and what side of the street or walkways relevant objects are placed. Notes: This attribute defaults to none to keep for low polygon counts. The rule that this attribute controls attempts to place a standard stop light at the corner of the street near the edge of the sidewalk shape so that is serves the lanes adjacent to the curb. Both will put them at both sides of the street. **Updated Rules Support @Handles for translation for Traffic Lights.**
Sidewalk_Way_Finder	Default Value: false Description: Will create a Way Finder near a bus stop. Notes: Within the bus stop geometry allocation (40 ft. long split), there are splits that will a way finder if this is turned to true and a bus stop is appearing in the rule.
Sidewalk_Bike_Rack	Default Value: false Description: Will create 2 bike racks near a bus stop. Notes: Within the bus stop geometry allocation (40 ft. long split), there are splits that will insert 2 bike racks if this is turned to true and a bus stop is appearing in the rule.

Parameter Group: Population

This parameter group will be discussed as a class rather than discuss each attribute individually because of their similarity. Each attribute will be mentioned, but only exceptions will not get detail.

Default Values: For most of the population objects that can be inserted (Vehicles, Buses, People, and Bikes), these setting are set to 0 at all LOD settings except Very High LOD as of the 2018 version of the rule.

Description: Generally, these attribute control how many 3D models representing road users are loaded on to the network. Many of the allocation attributes are in terms of Vehicles or Objects per KM which essentially loads objects to the asked for density based on evaluating the length of the lanes available. The rules that do this allocation are essentially Recursive Allocators (adapted code from previous rules) that will check to see if there is available space before inserting the relevant object at the appropriate density. The main one of interest if is Vehicles per KM, which has two fractional components that play into it, Taxi Percentage and Mixed Traffic Bus Percentage. Bus Lane Buses per Km and Bikes per KM only create road users in *the relevant lanes* they are intended for (Bus Lanes for buses and Bike Lanes for Bikes), so these attributes do nothing if there are none of the relevant lanes. If these percentages go up, so do the number of taxis and mixed buses as part of this allocation. The only other population attributes expressed in terms of percentages are Parked Car and People percentages. Parked Car Percentages maxes out at 1 (all parking spaces allocated), but people percentage will keep growing if put past 1. Generally, the higher the percentage, the higher the number of people loaded onto sidewalks.

Special Note about Parked Car Angle	Hidden Attributes (Reveal by Deleting @Hidden Annotation) There are two hidden attributes within population that relate to the on-street parking vehicle insertion (Right_Parked_Car_Angle and Left_Parked_Car_Angle. These attributes control additional customization of the car angles for angled parking. So, changing this attribute will change the angle at which cars are "parked" in the rule.

Parameter Group: Custom Objects

This parameter group will be discussed as a class rather than discuss each attribute individually because of their similarity. Each attribute will be mentioned, but only exceptions will not get detail.

Default Values: A series of strings that are file references to specific object files in the assets folder.
Description: Make sure object/3D files are aligned so that UP is aligned to the Y Axis. Keep in mind objs were inserted assuming a standard alignment.
Notes: A lot of scope casting is done within the street rule to make sure that the vertical direction (up and down relative to the street) is set to the Y axis. This means that if you have a 3D sketch up model where up is along the Z axis, and you want to use it in the rule then you need to change the model and save it before insertion. There are other OBJs within the asset folder to choose from if you don't like the ones shown as defaults.

Parameter Group: Bridges

Bridge_Display	Default Value: "Off" Description: Determines the various conditions by which the bridge rule will turn on. **In the 2018 version of the rule, the Occlusion functions that were used for collision detection are removed per the request of Esri**. Notes: Something important to mention about the Bride Rule code is that it is largely sourced from code from Chris Wilkins. The main modifications made to it are to some of its options to support cross sections and changes required to integrate it into the rule. In the case of cross sections, the option for "Extrusion Only" is helpful for making nice cross sections for presentations and standalone street segments. Just something to keep in mind. **Updated Rule:** Display settings for "Dimension Extrusion" will add dimensions splits to the underside of the extrusion.
Bridge_Starts_At	Default Value: 3 Description: Determines the threshold distance above the standard elevation for the bridge rule to trigger based on Elevation. Notes: There are some occlusion functions that will try to figure out where the ground is or where the piers collide with objects depending on the setting. However as noted above, you might have some errant behavior and unpredictable performance in this function. This attribute determines the elevation at which bridge would be triggered by the rule and can likely be a mapped attribute during a project (so it can be calculated in an ArcMap setting).
Bridge_Thickness	Default Value: 1

	Description: Determines how thick the supporting cement structure of the bridge is. Notes: Thickness of the bridge is controlled by this attribute.
Pier_Distance	Default Value: 23 Description: Determines the distance between Piers. Notes: Distance between each of the cement piers that reach the ground is determined by this attribute.
Pier_Width	Default Value: 2.3 Description: Determines the width of the Piers. Notes: Thickness and width of each pier is modified by this attribute.

Parameter Group: Trees (Imported from ESRI.lib)

Special note: The Tree attributes at the bottom of the rule are from the imported Tree rule. The way this imported rule interacts with the street rule is important to understand, especially if you want to change the height and radius of the imported models. The attributes here give more fine control over the trees, and a brief description of each attribute and how it relates to the rule will help provide some idea of how change how vegetation is used.

The most important thing to know about how trees work in the rule is the following: **the model choice** is based on the trees name, and that is chosen by the Sidewalk Tree Type and Median Tree Type settings. However, the **size and radius** of the inserted tree objects is determined by the tree type within the tree section of the rule. In order to have the tree models match the size and radius there are supposed to have, **Copy the** name of the sidewalk or median tree choices you have made while segments are selected, and then **Paste the** name of the tree into the "Name" Attribute. Their sizes and radius should then set to a default size and radius for that plant type, while still being adjustable to meet the user's needs. Generally, you can manually change the Height and Radius attributes to match what you would like to see for your plantings. There is code in the rule for it adjust automatically if two comments are deleted in the "TreeInsert (Location, Percentage1, Tree_Type1, Tree_Type2)" rule, but this removes the ability to adjust radius and height manually. You can customize it as you see fit past that.

Relevant import code:

"import Tree: "/ESRI.lib/rules/Plants/Plant_Loader.cga" # Taken from ESRI.lib and uses its assets. Keep this in mind when using rule."

Name:	Default Value: "Orange Tree" Description: As mentioned in the special notes, this attribute does not control the model choices, but controls the default size and radius (defaults to the average size for that species) for the tree selection done on the selected shapes. If you want to have differently sized trees for the sidewalks compared to the medians, it is suggested that you select the **sidewalk and street start shapes *separately*** and then change their attributes. The default is set to an Orange Tree here, because it is a typical height and radius expected for a good small to medium street tree.
Height:	Default Value: 9 (~27 ft.) Description: These attributes no long do anything in the 2018 version of the complete street rule. Defer to the Sidewalk/Median
Radius:	Default Value: 2.5 (~7-8 ft.) Description: As mentioned in the special notes, this attribute controls the radius of the trees in the complete street rule. You can adjust this manually to make tree models wider or narrower, but what is happening is the scope in the x-z direction is just being adjusted.

Representation:	Default Value: "Model" Description: Enables the choice between a 3D models, fan model, or an analytical model choice for the tree insertion. Fan model is a great way to reduce the polygon counts for web scenes.
Transparency:	Default Value: 0 Description: This attribute controls the transparency of the trees.
OverwriteColor:	Default Value: "" Description: This attribute is used to overwrite the colors of the tree models and color them to some color chosen by the user. It defaults to doing nothing.
RandomRotation:	Default Value: true Description: This attribute will randomly rotate the trees so that when they are inserted, they will seem more natural (not all having the same exact orientation).
RandomBrightness:	Default Value: false Description: This attribute will create variation in the "brightness" of the trees, making them seem different colors. This can have a slowing effect on web scenes because it can increase the amount of instancing caused by trees.
RandomHeights:	Default Value: Mature and Young Description: This function should allow some default randomization of heights but might not work depending on the setting choices done within the street rule.
Reports:	Default Value: "None" Description: Activating the various options for this attribute turn on various reports related to either various attributes of the trees or instance information related to how the objects were inserted.

Reports Discussion

Reports are dynamic analytics that allow a CityEngine user to generate numerical tabular reports based on "rule-based calculation and accumulation of a model's parameters" (CE Reporting Tutorial, 2015). There are more materials on what reports are and how they work, but the most important thing to know about they are that they allow real time evaluation of models based on any type of arbitrary criteria. In the street rule, the main report types that are discussed below are object counts, arbitrary geometry reporting, paint reporting, braking sight distances, and multimodal segment evaluations.

Updated Rule Notes: All the reports in the rule have technically been changed to use "." instead of ":", to support the report nesting structure required for dashboards. This means that the reports have been effectively "grouped" into related categories are support analytics related to each design made with the street rule. In addition, some reports have been removed, and some have been added. The main new report is mode area, which is discussed in detail below. Only some of the changes in the code and reporting is documented because it is subject to change, but the main changes are documented below for Modal Preference and aspects of the geometry reports. In the most updated versions of CityEngine reports are fully integrated into the Dashboard Settings.

Arbitrary Geometry Reports

- **Description:**

- These consist of a series of reports on different aspects of the geometry of the street including widths of different lanes, area of center sections, area of parking spaces (N is the number of parking spaces), and the area of the pervious surfaces (vegetation area). These reports vary, and there are different ones not shown in the image below. Just learn to read them, and keep track of the Sums, Averages, and counts (N). All have useful information.

- **Example Images:**

Center: Center Section Area	1	0.00	442.81	0.00	442.81	442.81	442.81	0
Center: Median Walkway Area (m^2)	20	0.00	180.65	0.00	9.03	8.77	9.97	0
Cut/Fill: Total Median Cut/Fill Volume (m^3)	1	0.00	45.17	0.00	45.17	45.17	45.17	0
Lane-Transit: Transit Lane Area (m^2)	2	0.00	424.19	0.00	212.09	212.05	212.14	0
Lane-Transit: Transit Lane Width (ft)	2	0.00	22.00	0.00	11.00	11.00	11.00	0
Lane: Actual Lane Width (ft)	2	0.00	21.65	0.00	10.83	10.83	10.83	0
Lane: Asphalt Only Area Total (m^2)	26	0.00	724.76	0.00	27.88	1.52	208.78	0
Lane: Car Lane Area (m^2)	2	0.00	417.51	0.00	208.75	208.73	208.78	0
Parking: Left Parking Space Area (m^2)	9	0.00	133.45	0.00	14.83	14.77	14.87	0
Parking: Right Parking Space Area (m^2)	9	0.00	133.44	0.00	14.83	14.76	14.87	0
Parking: Total Parking Space Area (m^2)	18	0.00	266.89	0.00	14.83	14.76	14.87	0
Vegetation: Construction, Tree Cost	10	0.00	7500.00	0.00	750.00	750.00	750.00	0
Vegetation: Pervious Area	10	0.00	240.86	0.00	24.09	23.40	26.58	0

- **Example Code:**
  - report ("Parking: "+Side+" Parking Space Area (m^2)", geometry.area)
- Updated Rule Notes:
  - Generally, these reports were minimally changed, and some are in the process of being removed because of the lack of added value some provide compared to some of the other reports. Many of the Lane Reports as an example were removed and replaced with “Mode Area” reporting.

Braking Sight Distances

- **Description:**
  - Stopping sight distance is a common type of stopping distance used in street design. It represents the “near worst-case distance a vehicle driver needs to be able to see in order have room to stop before colliding with something in the roadway, such as a pedestrian in a crosswalk, a stopped vehicle, or road debris. Insufficient sight distance can adversely affect the safety or operations of a roadway or intersection (Wikipedia, 2015).” The reporting follows the general methodology discussed by AASHTO, borrowing the assumptions they make for constants such as Brake Reaction Time and Deceleration rate. The design speed is also a constant and not a controllable attribute and is abstracted away. You can change it in the code. Generally, it will be equal to the Speed Limit if the Speed Limit is less than 40, but design speed will be equal to the speed limit +7.5 if the Speed Limit is greater than 40.
- **Example Image:**

Speed:Level Braking Distance (ft)	1	0.00	64.88	0.00	64.88	64.88	64.88	0
Speed:Level Braking Reaction Distance (ft)	1	0.00	95.55	0.00	95.55	95.55	95.55	0
Speed:Level Stopping Sight Distance (ft)	1	0.00	160.43	0.00	160.43	160.43	160.43	0

- **Example Code:**
  - **const Brake_Reaction_Time = 2.5 #s**
  - **const Deceleration_Rate = 11.2 #ft/s^2**
  - **const Design_Speed = case Speed_Limit_in_MPH>=40:Speed_Limit_in_MPH+7.5 else: Speed_Limit_in_MPH**
  - **const *Brake_Reaction_Dist*= 1.47**Design_Speed***Brake_Reaction_Time***
  - **const *Braking_Dist* = 1.075*((*Design_Speed*Design_Speed*)/*Deceleration_Rate*)**
  - **const *Stopping_Sight_Dist* = *Brake_Reaction_Dist+Braking_Dist***

- **report ("Speed:Level Braking Distance (ft)",Braking_Dist)**

Multimodal Segment Evaluations

- **Description:**
  - These multimodal preference metrics are largely measures of multimodal design evaluations that rank streets based on how well they accommodate each mode type. Bicycle stress is actually adapted from a federally funded study, and the other metrics are best on the rule author's best judgement based on various literature sources. All the metrics are normalized from 0 to 1, with 1 being better for that mode and 0 being worse for that mode. They are on a rank from 0 to 1 to make incorporating it into to the thematic displays, but also to give all the metrics a common scale. **These analytics can and should be modified if you have a great idea for evaluating street design.** For example, future ideas might be a freight accommodation score based on the width of the curb lanes, parking availability (for loading and unloading), speed limits, and other aspects of the streets design. It is actually a lot of fun to think about, because it is custom procedural real time design and evaluation. **In the updated rule, the only change was that the metrics were renamed to "Modal Preference", but the underlying metrics stayed the same. Miscellaneous bug fixes were applied to bike metrics to handle cycle tracks and parking more appropriately.**
- Special Note on Bike Stress:
  - Just to be clear the bike stress metrics are based on study which designed a set of criteria to evaluate streets for how they function in a low stress bicycle network. In the rule, a 0 corresponds to an LTS 4 street, .33 and LTS 3 Street, .66 to an LTS 2 street and a 1 is a LTS 1 street. It provides a great case study on how street rules can be applied to evaluate design alternatives in real time
- **Example Image:**

LTS (0 to 1 scale):Auto Stress	1	0.00	0.20	0.00	0.20	0.20	0.20	0
LTS (0 to 1 scale):Bicycle Stress	1	0.00	0.66	0.00	0.66	0.66	0.66	0
LTS (0 to 1 scale):Pedestrian Stress	1	0.00	0.80	0.00	0.80	0.80	0.80	0
LTS (0 to 1 scale):Transit Stress	1	0.00	1.00	0.00	1.00	1.00	1.00	0

- **Example Code:**
  - See _BikeRank, _AutoRank, _PedRank, and _TransitRank in the rule.
  - report("LTS (0 to 1 scale):Bicycle Stress", _BikeRank)
- **Updated Rule Example Image:**

Modal Preference (0 to 1).Auto	2	25.00	0.80	13.33	0.40	0.40	0.40	0
Modal Preference (0 to 1).Bicycle	2	25.00	2.00	33.33	1.00	1.00	1.00	0
Modal Preference (0 to 1).Pedestrian	2	25.00	1.60	26.66	0.80	0.80	0.80	0
Modal Preference (0 to 1).Transit	2	25.00	1.60	26.66	0.80	0.80	0.80	0
Modal Preference (0 to 1)	8	100....	6.00	100....	0.75	0.40	1.00	0

- **Updated Rule Example Code:**
  - See _BikeRank, _AutoRank, _PedRank, and _TransitRank in the rule.
  - report("Modal Preference (0 to 1).Bicycle", _BikeRank)

Mode Area

- **Description:**
  - These reports provide the area of the entire street that is dedicated to each mode. These reports report on 5 mode area types on the street and they are Bicycle, Pedestrian, Transit, Auto, and Conflict Zones (cross walks and some other mixing areas). Generally, bike lanes are included in the bicycle area reports, sidewalks for pedestrians, transit lanes for transit, and through lanes for autos. These reports are designed to be used with dash boards to illustrate changes in area dedicated to each mode and to communicate how changes in a street can serve more users.

- Example Image:

Mode Area (m^2).Auto	28	15.90	2610.02	34.96	1305.01	1305.01	1305.01	0
Mode Area (m^2).Bikeways	94	53.40	1151.70	15.43	575.85	575.85	575.85	0
Mode Area (m^2).Conflict Zones	8	4.54	259.20	3.47	129.60	129.60	129.60	0
Mode Area (m^2).Pedestrian	4	2.27	2477.29	33.19	619.32	619.32	619.32	0
Mode Area (m^2).Transit	42	23.86	965.71	12.93	482.85	482.85	482.85	0
Mode Area (m^2)	176	100....	7463.92	100....	621.99	129.60	1305.01	0

- Example Code:
    - **report("Mode Area (m^2)."+str(Usage),geometry.area*Area_Fraction)#Area Fraction avoids double counting some shapes**

Curbside Allocation

- **Description:**
    - As curbside management has increasingly been a heavily discussed topic in transportation, the rule now includes reports on how the parking lane is allocated. These reports are linear allocations in meters of how the parking lane is being used among a set of different allocations include passenger drop-off, dockless bikeshare/scooters, freight loading zones, parklets, and parking.
- **Example Image:**
    -

Curbside Management	8	100.00	67.09	100.00
Curbside Management.On-Street...	6	75.00	40.20	59.92
Curbside Management.Passenger...	2	25.00	26.89	40.08

- **Example Code:**
    - **report("Curbside Management.Sidewalk Cafe (m)",geometry.du(0,unitSpace))**

Handles

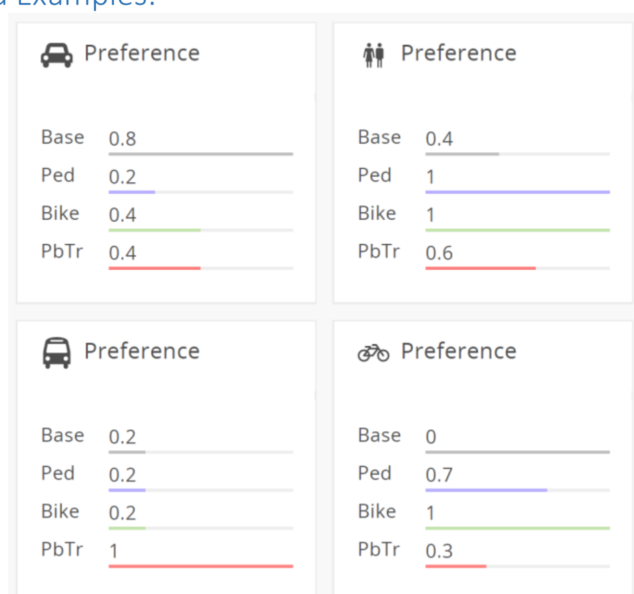
Handles are new features to the complete street rule, with only the most demanded handles supported at this time. So far this includes handles to adjust the height of trees (0 removes them), and to translate certain objects like benches, traffic lights, and streetlights. If you want to request a new handle you can contact me or file a request on Github. In order to view most handles related to most street objects, must have their hidden parameters revealed by commenting out the hidden annotations starting in the “Handles” section of the rule. You can find the parameters that need to be unhidden by searching for ‘Group(“Handles”)’ in the Complete_Street_Rule.cga file.

Figure 3 Example of Tree Handle Adjusting Height of an Individual Tree

Dashboards

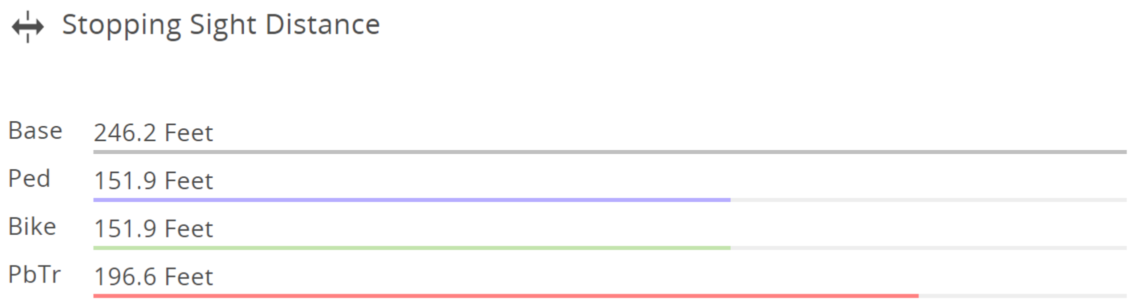
With the advent of CityEngine 2015.2, analytical Dashboards were included as a new feature of CityEngine. While the documentation page for Dashboards provide a good explanation of how to use them, this section will include some discussion on how they apply to the Complete Street rule and what some key reporting dashboards might be. The dashboards shown here use the street shown below as potential examples (more recent versions of CityEngine will have Dashboards that focus on scenario enablement, you can create similar dashboards however like those below). As of 2020, a dashboard sample is provided with multiple scenarios for the Complete Street Rule, and includes sample markdown blocks describing the different metrics shown.

Mode Preference Dashboard Examples:



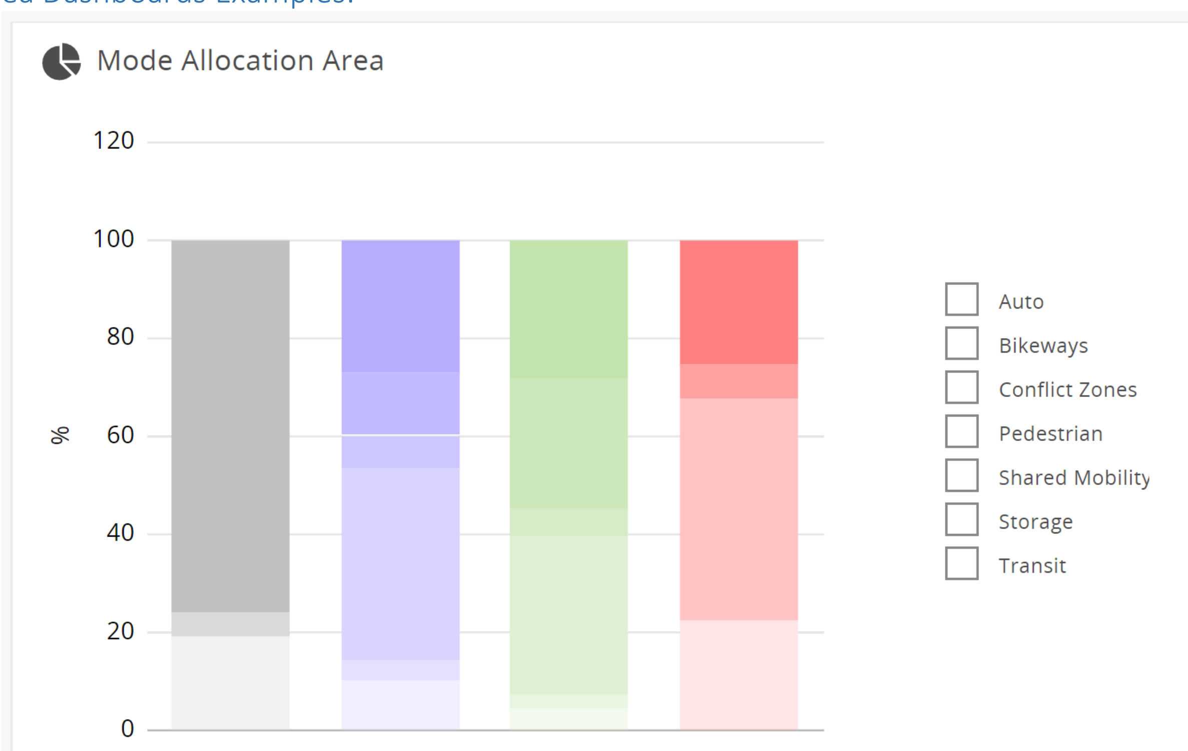
For mode preference a good default is to defer to the bar and pie charts to show which modes are served appropriately by a street. Something to note about the pie char tis that the inner rings allow you to change rule parameters and see how they change the relative preferences of different modes.

Speed and Stopping Distances Dashboards Examples:



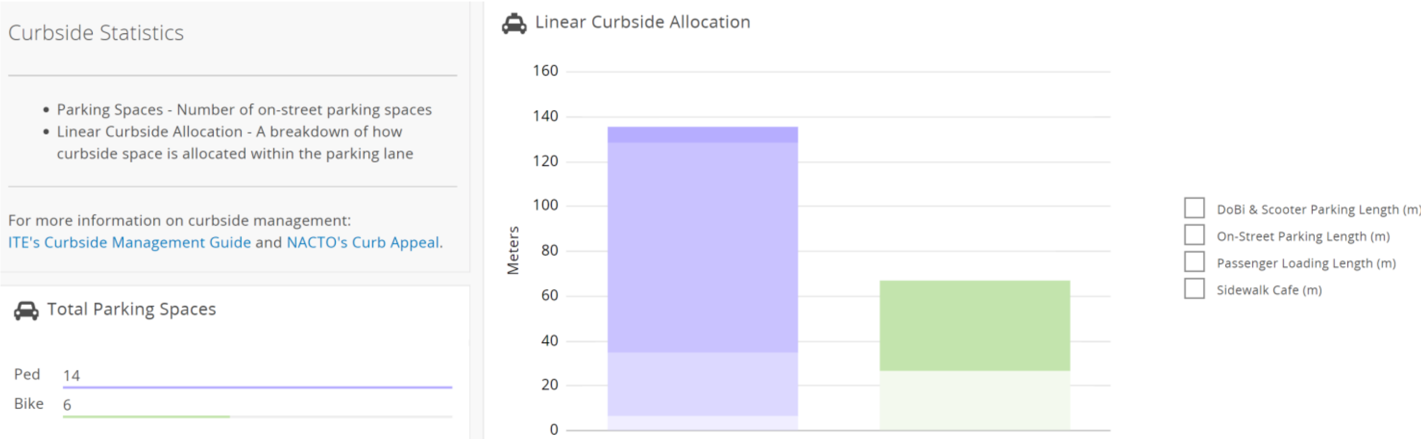
The speed dashboards are best with bar charts/single numbers and use the reporting to show how stopping sight distance changes based on changes in the posted speed. See Posted Speed Parameter for details on how this is used to determine design speed.

Mode Area Dashboards Examples:



These dash boards also work best with the column and bar chart dashboards, but the statistics chosen to depend on what you wish to communicate. In the case of this cross section they are set to Sum for the bar charts and percentage for the pie chart.

Curbside Allocation Dashboards



These dashboards can report how the parking or curbside lane is used by linear meter. Can be used to provide bar charts by scenario. Only activates for streets with parking lanes of some type chosen.

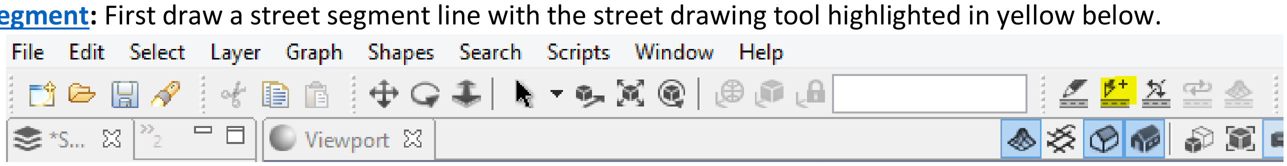
Street Trick Notes

This section provides a few tips on how to do some common tasks with the street rule or interesting street designs.

Cross-Section Creation and Image Taking

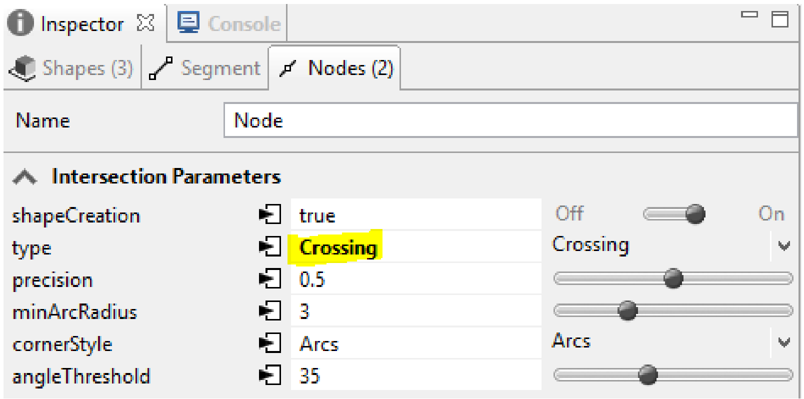
A cross section like the one shown below can be made pretty rapidly with the street rule, and it becomes a faster process the more familiar and organized you do about creating segments. There are a few key steps you have to follow in terms of changing default settings, and some tips about creating images. Below outlines the general process for creating cross section images such as this one. The links provide references to the relevant CityEngine documentation.

1. **[Draw Segment](#)**: First draw a street segment line with the street drawing tool highlighted in yellow below.

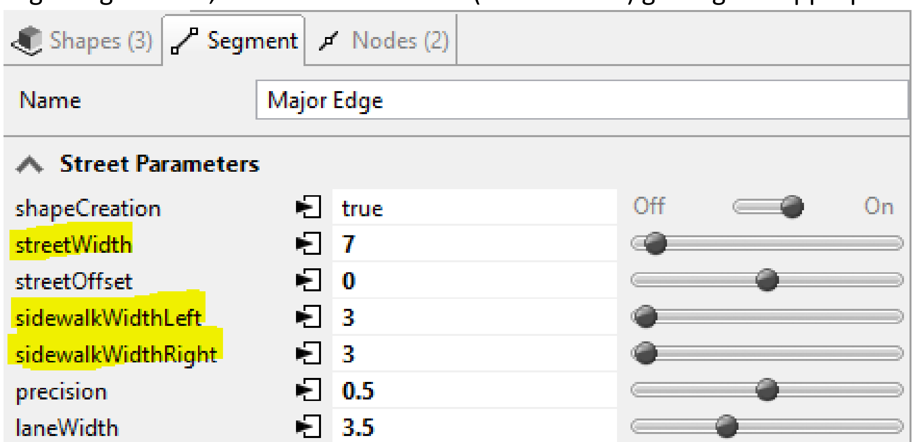


-

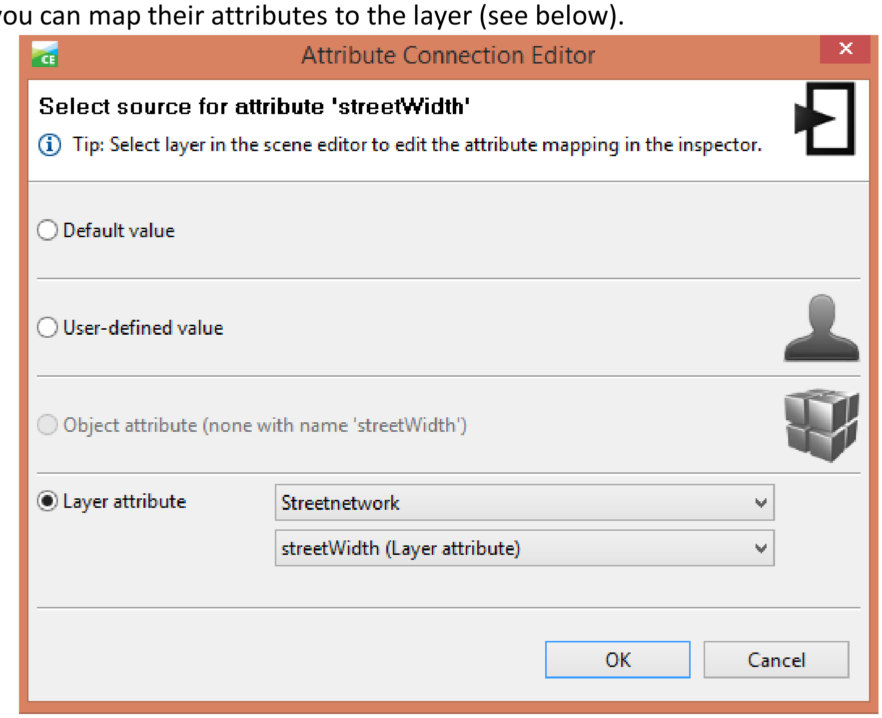
2. **[Change Street Parameters](#)**:
   - By default, a lone segment will be given a roundabout at both ends, but for a cross section we want to change the street parameters for the shapes so they are set to “Crossing”. So, select then entire street segment, and its nodes (points at the end). Change the nodes parameters to “Crossing” from Smart.

i. 

- Then make the street Sidewalks + Street width equal to the width of your right of way you want to create a cross section for. Give them the appropriate sizes for each street parameter, with right and left sidewalks getting a width, and the street width (curb to curb) getting the appropriate width.

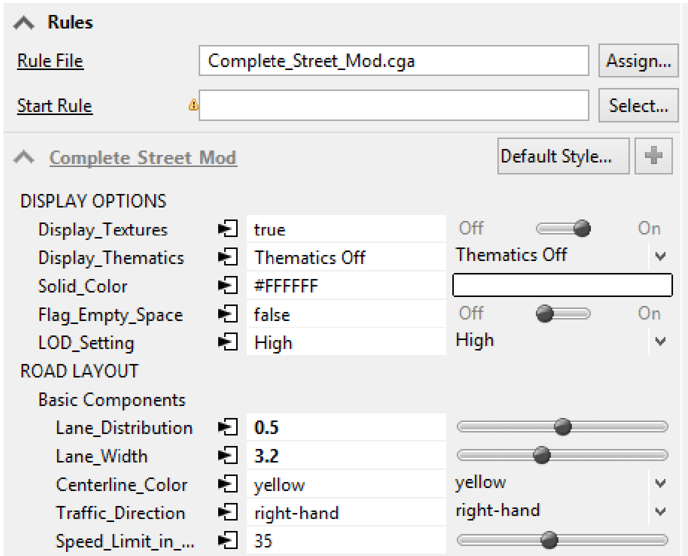
i. 

- If you have to do this for a lot of segments whose data is from a GIS data base of centerlines, make sure their break points are at intersections, and calculate the street width and sidewalks width within ArcGIS. Then you can map their attributes to the layer (see below).

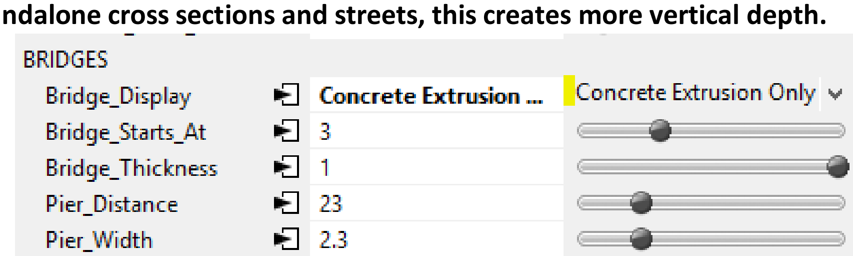
i. 

3. [Change Relevant Street Rule Attributes](#):

- At this stage you have a street segment that is ready to have the rule applied to the segment and then generated. The default will be largely travel lanes, and because it is an isolated segment have no cross walk, stop bar, and a negative stop bar setback (it assumes it is connecting to another street segment). If you want crosswalks and stop bars, activate them. The link above shows an example of how bike lanes are added to a street segment with the rule as an example.

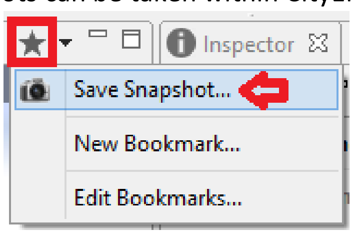
    i. 

- **For cross sections a suggested attribute to change is the Bridge Display to "Concrete Extrusion Only". For standalone cross sections and streets, this creates more vertical depth.**

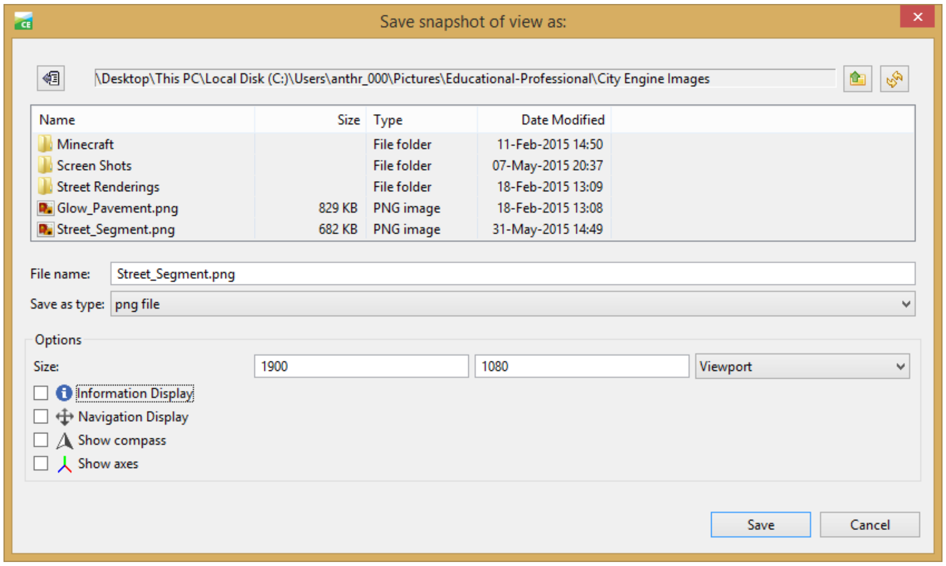
    i. 

4. Take a [Snap Shot](#) or [Export a Web scene](#):

- Snap Shots can be taken within CityEngine by clicking on the star, and then selecting Save Snapshot.

    i. 

- For Cross sections it is suggested you take an image of high resolution (at least 1080), and then downsized in Photoshop or GIMP. In addition, turn off ALL display options so that various CityEngine Display assets do not appear in your image. In addition it helps to toggle the visibility of the **[segment lines layer](#)** when the cross section image is taken. Also, it might help to turn on the [shadows and ambient occlusion in the scene settings](#).



i.

- Exporting a web scene has documentation and tutorials so it will not be covered here. Just keep in mind that you want to keep your polygon count to a moderate level for most web scenes.

Creating Curb Extensions:

- Making curb extensions was discovered on accident, and they are by no means perfect. However, you can create three segments and give the two end one's different sidewalk widths to approximate curb extensions. The gap between the streets is the "Joint" start shape that as mentioned above is sent to asphalt.

Other Forum Discoveries:

- Again you can find out more things at the original Feedback forum on Geonet.

Removed and Deprecated Features Archive

The section below documents features of the rule that are not part of the current release and are just clipped from previous portions of the documentation.

Paint Reports

This parameter group was removed from the rule in 2018. The description is maintained for the group at the end of the documentation.

Default Values: $1.6 Dollars per square foot

Description: Useful Installed costs/sq. ft.: Paint $1.6, Epoxy/MMA: $8-11, Thermoplastic $10-14, Colored pavement: Varies. See comments for details-(Cntr+F NACTO Urban Bike Design Guide). Costs can become dated due to inflation, change as needed.

Notes:

This entire class of attributes feeds into the paint cost reporting. It turns square footage of paint applied into a cost estimate with a simple multiplication. As a result applications that such a symbols that might have per application costs might need different treatments, but it does provide a rough ball park estimate for different applications based on the cost multiplier chosen (based on the average cost per square foot). The comments that provide details on some background on the source for the paint cost reporting estimates.

“Default values taken from NACTO Urban Bike Design Guide, 2nd edition sees comments for details. Costs can become dated due to inflation, change as needed. -DJW

- Paint- pigment and binder, low durability (6 months-2 years based on conditions), low traffic
    - Material cost: $.06 Sq. Ft. raw material/ $1.20-$1.60 Sq. Ft. installed.
- Epoxy- epoxy/resin, moderate durability impacted by pavement quality(3-5 years), moderate traffic
    - Material cost: $1-$3 Sq. Ft. raw material/ $8-$11 Sq. Ft. installed.
- MMA- acrylic based resin, moderate durability impacted by pavement quality (3-6 years), moderate traffic
    - Material cost: $3-$4 Sq. Ft. raw material/ $8-$11 Sq. Ft. installed.
- Thermoplastic- polymer resin, pigment, beads, filler- moderate-high durability (5 years or longer)
    - Material cost: $3-$6 Sq. Ft. raw material/ $10-$14 Sq. Ft. installed.
- Colored Pavement- bituminous pitch, sand/gravel/pigment-very durable, last as long as typical asphalt depending on conditions
    - Material cost: Pigmented asphalt typically costs 30-50% more than non-colored structural asphalt, thin overlay applications have varying costs.”

Paint Reporting

- **Description:**
    - All of the paint cost reporting and paint area reporting is going through two main reporting operations with a complicated series functions/constants controlling the process. Generally, if there is a texture with a single color, the entire area of that particular tile will be reported. However, in the clear case where there are textures that obviously have a mixture of asphalt or different color paints, the fraction of each color in the texture (calculated in GIMP) is used to determine the area fraction that modifies the area calculation. In the case where two paint colors are on the texture, the assumption is that the second color (usually white) is painted over an already painted surface. However, this may not the case for some type of paint applications such as preformed thermal plastic applications, where the surface might only be made up of the area it is formed for.
- **Example Image:**

Paint Cost Estimate: green Painted Area ($)	14	0.00	3915.08	0.00	279.65	30.69	840.96	0
Paint Cost Estimate: red Painted Area ($)	14	0.00	6862.71	0.00	490.19	107.54	1337.21	0
Paint Cost Estimate: white Painted Area ($)	44	0.00	1140.01	0.00	25.91	1.25	110.72	0
Paint: green Painted Area (m^2)	14	0.00	227.33	0.00	16.24	1.78	48.83	0
Paint: red Painted Area (m^2)	14	0.00	398.48	0.00	28.46	6.24	77.64	0
Paint: white Painted Area (m^2)	44	0.00	66.19	0.00	1.50	0.07	6.43	0

- **Example Code:**
    - **report("Paint Cost Estimate: "+paintColor+" Painted Area ($)",(geometry.area*Area_Fraction*SquareFeet)*_PaintCost(paintColor))**

Object Counts

- **Description:**
    - These reports are very simple, any time an object is inserted, and it is counted in the report by a number incrementing by one per object. Often these reports are done with the same line of code, but have strings passed to their functions that describe their type and thus inform how the report is shown. **Partially Removed in 2018 Version of Complete Street Rule, some object reporting remains, but most were removed.**
- **Example Image:**

Objects: Bench Count	10	0.00	10.00	0.00	5.00	5.00	5.00	0
Objects: Bike Rack Count	4	0.00	4.00	0.00	2.00	2.00	2.00	0
Objects: Bus Stop Count	2	0.00	2.00	0.00	1.00	1.00	1.00	0
Objects: Parking Meter Count	18	0.00	18.00	0.00	9.00	9.00	9.00	0
Objects: Street Lamp Count	20	0.00	20.00	0.00	6.67	5.00	10.00	0
Objects: Traffic Lights Count	2	0.00	2.00	0.00	1.00	1.00	1.00	0
Objects: Wayfinder Count	2	0.00	2.00	0.00	1.00	1.00	1.00	0

- **Example Code:**
    - report("Objects: "+Object_Type+ " Count",1)

Removed Parameters

Centerline_Color	Default Value: _getInitialCenterline Description: Choses color for the centerline if there is a center line. This attribute does nothing to other center types. *Removed in 2018 version of the rule, replaced with center space settings entirely.* Notes: The default value is a worker function _getInitialCenterline. This function just makes sure no center line is allocated to one-way streets. The current functionality here is inherited and modified by the Street_Modern_Standard.CGA rule. Currently the none option just allocates a different texture, so if you want to actually make no centerline, change the Center_Type to Median, and make sure the center width is 0.
Flag_Empty_Space (Removed in Updated Rule)	Default Value: false Description: When this attribute is true, any time there is unallocated drainage space that is wider than 1/3 the current lane width (usually 3-4 feet), it will flag them as red. It helps find errant streets. Notes: In future versions of the rule, this attribute was removed for Space_Management.

Questions and Feedback:

I made this documentation on my own time, but even though I no longer work at ESRI I will try to answer questions and receive feedback about the street rule on Geonet or linked in.

My Geonet account is here: https://geonet.esri.com/people/Holisticbynature

My Linked-in is here: https://www.linkedin.com/in/david-j-wasserman/

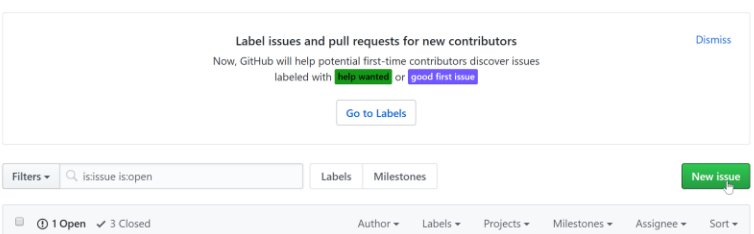
My Github is here: https://github.com/d-wasserman

Feel free to reach out if you have questions. I hope to make modifications to this document in the future as changes to the rule might be made, but this is a project of my own time and energy so bear with potentially slow update periods.

Feature Requests

If you want to file a feature request for the rule or have feedback you are fine with sharing publicly, you can file an Issue over GitHub pretty easily.

The stops are as follows:

1. Create a GitHub Account if you don't have one.
2. Go to the Complete Street Rule Issues Page: https://github.com/d-wasserman/Complete_Street_Rule/issues
3. Click on "New Issue"
    a. 

4. Fill out a description and title for the issue, label it as appropriate, then hit submit.